FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

A Framework for Distributed Intelligence for Energy Efficient Operation of Smart Homes

DISSERTATION

submitted in partial fulfillment of the requirements for the degree of

Doktor der technischen Wissenschaften

by

Christian Reinisch
Registration Number 0026411

to the Faculty of Informatics
at the Vienna University of Technology

Advisor: Ao. Univ.Prof. Dr. Wolfgang Kastner

The dissertation has been reviewed by:

(Ao. Univ.Prof. Dr. Wolfgang Kastner)	(Univ.Prof. Dr. Ardeshir Mahdavi)

Wien, 13.11.2012

(Christian Reinisch)

Erklärung zur Verfassung der Arbeit

Christian Reinisch
Margaretenplatz 2/28, 1050 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit - einschließlich Tabellen, Karten und Abbildungen -, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

(Ort, Datum) (Unterschrift Christian Reinisch)

Abstract

The world-wide energy demand and with it the greenhouse gas emissions are rising constantly. One of the main energy consumers are buildings, both in the commercial and residential area. Regardless if viewed from an international, European or Austrian vantage point, the consumption figures call for immediate actions also in the building sector to reach emission goals such as the European Union's 20-20-20 targets. The requirements concerning energy efficiency in households are considerably different to those in commercial environments, hence a differentiated strategy must be developed for both of them. The huge number of households, the ever increasing count of energy consuming devices in the homes and the complex relations between energy efforts and the expected environmental effects make them a promising candidate for significant improvements.

In this context smart homes become increasingly popular as they offer improvements, such as increased comfort for their inhabitants at promised energy reductions. Nevertheless, in many cases smart homes cannot fully realize their promises, their systems are still complex to use, seldom tailored to pervasive energy savings and often not characterized by real smartness.

Thus, a novel system concept for smart homes is developed from scratch in this dissertation. The residential area has special characteristics and introduces specific requirements for any control approaches. Most obvious, comfort plays a main role for humans, and mostly energy efficiency comes only second after it. Hence, focus in the system design is given to the realization of energy savings while fully preserving comfort parameters of the users. For this purpose a comprehensive system concept for energy efficiency and comfort in residential homes is developed. It is realized as a multiagent system that implements the main smart home characteristics such as adaptivity, reactivity and proactivity in a dedicated and extensible software framework. It is capable of controlling its environment by exploiting automation technology, adapts to its users, alleviates the users of routine tasks and supports the overall energy efficient operation of the smart home. The design starts with an extensive requirements analysis, follows an established design methodology and results in a detailed specification of a multiagent based smart home control system. Finally, this system concept is prototypically implemented, tested and evaluated by means of simulation.

Kurzfassung

Der Energieverbrauch und mit ihm die Treibhausgas-Emissionen sind weltweit dramatisch im Steigen begriffen. Zu den größten Energieverbrauchern zählen dabei sowohl Wohn- als auch Bürogebäude. Bei Betrachtung internationaler, europäischer oder auch nationaler Energieverbrauchsdaten wird klar ersichtlich, dass zeitnah Gegenmaßnahmen besonders am Gebäudesektor ergriffen werden müssen, um Emissions- und Verbrauchsziele, wie sie die EU 20-20-20 Agenda vorsieht, erreichen zu können. Hierbei ist eine getrennte Betrachtung von Wohn- und Zweckbauten zielführend, da die an sie gestellten Anforderungen divergieren. Ein besonderes Optimierungspotential kann Wohngebäuden attestiert werden: sie existieren in großer Zahl und durch den intelligenten Betrieb der Vielzahl an Energieverbrauchern können signifikante Einsparungen erzielt werden.

In diesem Kontext stellen Smart Homes einen Lösungsansatz dar, der erhöhten Komfort für die Benutzer bei gleichzeitiger Energieeinsparung ermöglichen soll. Gegenwärtige Smart Homes können jedoch diese Versprechen nicht immer zur Gänze einhalten. Gründe dafür sind eine fehlende ganzheitliche Betrachtungsweise sowie die Komplexität der eingesetzten Systeme bei gleichzeitig fehlender Systemintelligenz.

In dieser Dissertation wird ein neuartiges Systemkonzept für Smart Homes entwickelt. Dieses ist auf die Anforderungen im Wohnbereich zugeschnitten und nimmt besondere Rücksicht auf die permanente Sicherstellung von Benutzerkomfort und Energieeffizienz. Der gesamte Design- und Implementierungsprozess ist daher auf eine ganzheitliche, umfassende Betrachtungsweise des Problemfeldes ausgerichtet. Die Designphase des Systems beginnt mit einer eingehenden Anforderungsanalyse und folgt etablierten Methodologien zur Spezifikation des Gesamtsystems. Die Umsetzung des Kontrollsystems für Smart Homes wird als Multiagentensystem vorgenommen, wodurch wichtige Systemeigenschaften, wie Anpassungs-, Reaktions- und Lernfähigkeit, realisiert werden. Das resultierende modulare Software-Framework nutzt vorhandene Automationssysteme zur Kontrolle der Umwelt und von Geräten im Smart Home, passt sich an seine Benutzer an und unterstützt diese beim energieeffizienten Betrieb ihres Heims. Abschließend wird eine prototypische Implementierung des Gesamtsystems vorgestellt, die mit Hilfe von Simulation getestet und validiert wurde.

Contents

Chapter 1

Introduction

In a study from 2006, the International Energy Agency (IEA) published the following facts [1]:

"*Today, energy use in residential, commercial and public buildings accounts for 35% of total global final energy consumption. Energy use in buildings increased 39% between 1973 and 2003 in IEA countries. Space heating is still the main end-use in buildings in these countries, but appliances are driving growth, while electricity is the most important fuel in IEA countries. [...] Accelerating progress to make energy use in buildings more efficient is indispensable. There is significant scope for adopting more efficient technologies in buildings. In non-IEA countries, the potential for improvement is even greater, as rapidly expanding economies offer enormous opportunities for investment in energy efficient technologies. Buildings and appliances account for about a quarter of the total CO_2 emission reductions below the Baseline Scenario in 2050 in the ACT (Accelerating Technology; author's note) scenarios.*"

Since then the situation has not drastically changed, neither in emerging countries nor in the most developed regions of the world, such as Europe or North America. In Austria, the consumption breakdown from 2010 [2] shows private households with a share of almost 26% in the total energy consumption. At the same time it becomes visible that the energy consumption of households increased from 1970 until 2010 by over 70%, now reaching more than 290 PJ[1] per year.

In Austrian households the energy consumption is distributed to the different domains as shown in Figure 1.1. According to Statistics Austria [3], most of this

[1] $Petajoule = 10^{15} Joule$

energy – almost three quarters – is consumed for heating, followed by household appliances/lighting and hot water production. Also in a general context of the overall Austrian energy consumption, still more than one quarter of the total useful energy is required for heating. Given these above figures, the need for immediate coun-

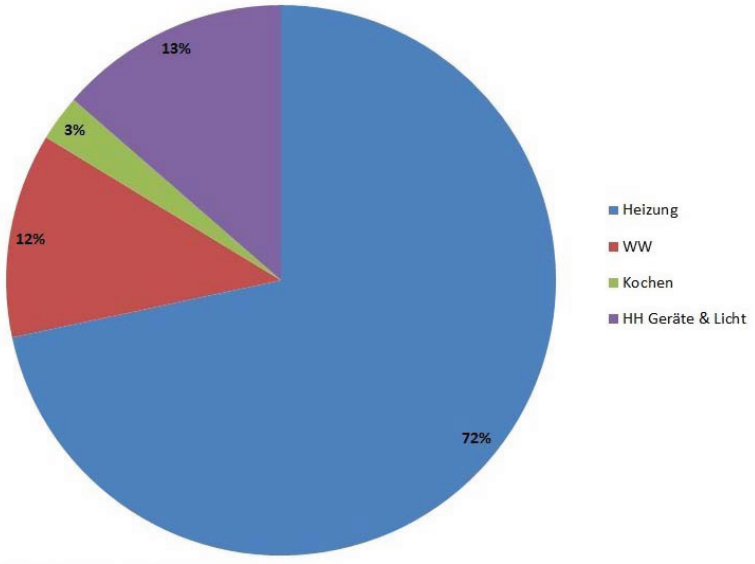

Figure 1.1: Energy distribution in Austrian households [4]

termeasures against the ever increasing energy consumption in buildings is quickly motivated. The time horizon for such mandatory reductions is influenced by the expected shortage of oil and gas availability in the coming decades as well as aggravated by the decision of several countries (e.g., Germany) to drop out of nuclear energy production following the Fukushima catastrophe in 2011. Meanwhile, the energy discussion has reached a wide political audience and energy efficiency becomes increasingly reflected in legislature. The European Commission has defined climate and energy targets (known as 20–20–20 targets) to be reached by 2020 by its member states. By then, a 20% reduction in European Union (EU) greenhouse gas emissions from 1990 levels, a raise to 20% of the share of EU energy consumption produced from renewable resources and a 20% improvement in the EU's energy efficiency shall be realized.

The 20–20–20 targets also highlight another immediate challenge. Facing the expected dramatic increase of the worldwide energy consumption in the future [5], it will not be sufficient to take action on the consumption side only. Rather, also

actions must be taken regarding the way how energy is produced to keep greenhouse gas emissions under control. Renewable energy sources are one of the means that tackle this problem on the producer side.

On the consumer side, the nature of the energy reduction mechanisms as well as their implementation strategies are still discussed controversially. On one end of the line, private households are a significant factor in energy consumption worldwide so that it is promising to implement conservation strategies there. On the other end, the end user is in most cases neither aware of his/her energy consumption (behavior) nor how energy can be saved most efficiently. One example is found in renewable energy sources whose efficient integration into the household's energy supply is a task of considerable complexity. This leads to either a suboptimal use of the available "free" energy or even a complete rejection of these technologies.

A key to sustainability in the residential building sector are automation systems and in particular smart homes. They provide technological means to increase energy efficiency while preserving comfort in the home. However, while smart homes have the potential to reduce the energy consumption in the home, currently available systems are not (technologically) mature enough to fully exploit all the potential that is theoretically available. It is thus the next logical step to develop the next generation of smart homes that contribute as much as possible to reaching the emission and energy efficiency goals already faced worldwide. The present dissertation aims at contributing to these ambitious goals by designing a novel smart home system concept that optimizes energy use, integrates renewable energy sources whenever possible and hence reduces the greenhouse gas emissions of the smart buildings.

The methodological approach of the dissertation is depicted in Figure 1.2. At the beginning an extensive literature study of state-of-the-art research as well as analysis of the related research fields building automation, home automation and smart homes are conducted. In particular, existing real-world smart home projects are discussed and analyzed regarding their functionalities and approaches. This research includes the identification of smart home characteristics, applications and current challenges and culminates in a new definition of a smart home. Based on this definition, the requirements of a smart home are identified and described in the following chapter. Emphasis is put on carefully addressing the smart home challenges that were previously identified. Furthermore, user stories that illustrate the daily operation of a smart home system are created to provide a deeper understanding of the smart home potentials. Guided by the user stories of smart homes, a first architecture of a

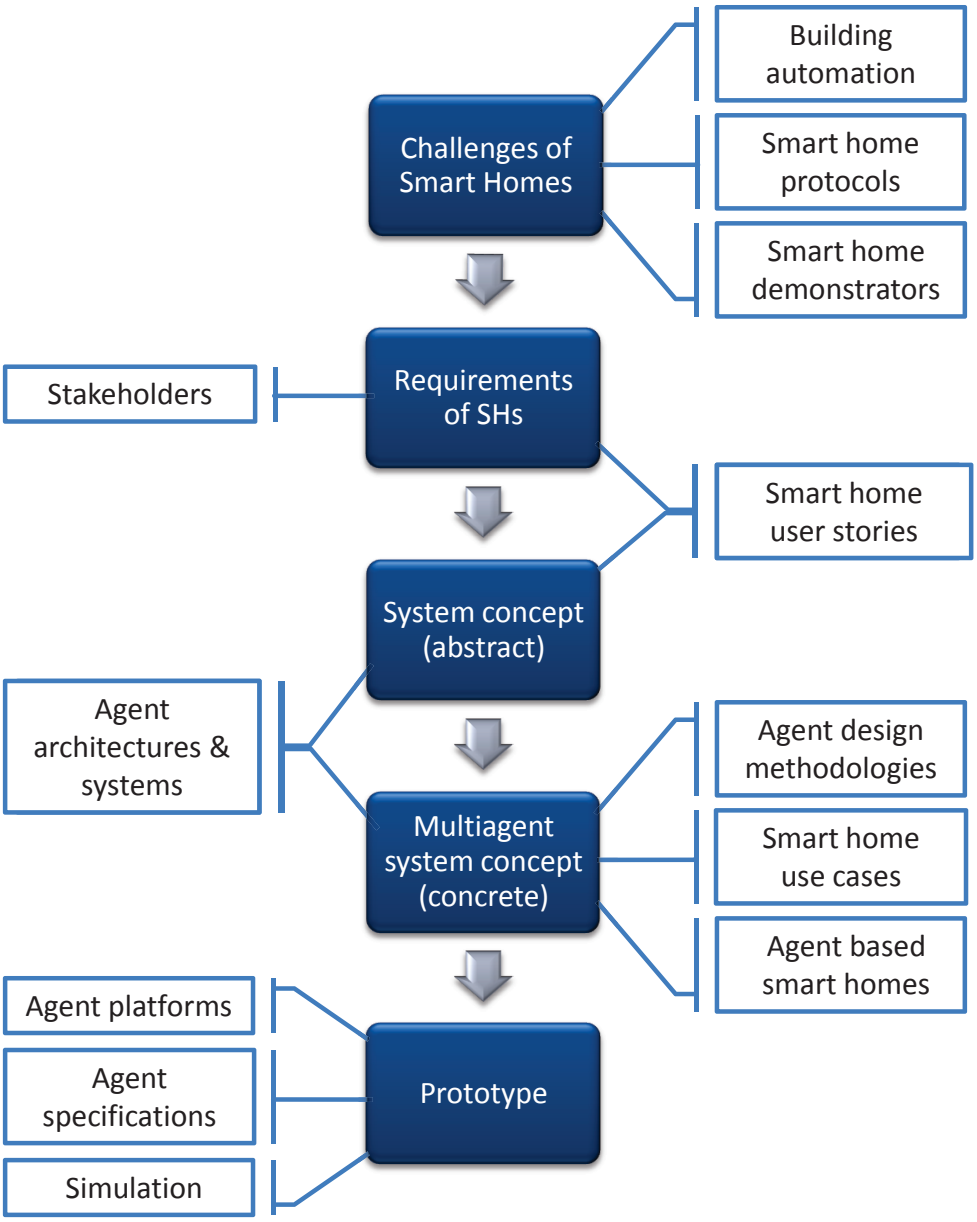

Figure 1.2: Methodological approach and influence factors

smart home system is developed that is capable of fulfilling the requirements as well as of addressing the challenges. Different system parts are identified and narratively described regarding their functionalities within the comprehensive system.

The system design chapter follows a classic software engineering process and has the goal of refining the smart home system concept. Therefore, first use cases that

specify all aspects of the smart home system behavior in detail are developed and formally described. Based on them, an implementation technology is selected. For this purpose, (technological) design and implementation approaches are explained and their applicability is motivated. In particular, the design of the smart home system based on a (multi)agent approach is carefully analyzed and justified with the help of criteria found in related agent research. This includes the selection of a particular agent architecture and paradigm. The multiagent based approach is furthermore compared to other agent based projects in the automation domain to identify proven approaches as far as applicable for the present smart home system.

The specification of the multiagent based smart home system is obtained by applying a selected agent system design methodology to the smart home use case. In order to identify the most suited design methodology, several relevant methodologies are discussed and evaluated prior to the specification process. The specification is influenced by many aspects and outputs already established or created previously, in particular the list of smart home requirements, the user stories and use cases as well as agent system characteristics and related projects. The specification consists of several design artifacts that capture parts of the system behavior and that represent the input for a subsequent implementation.

Such a prototype implementation closely based on the system specification is described in the prototype chapter. Previous to the implementation description, the used programming and system technologies are explained. With the help of simulation, the multiagent system prototype is evaluated regarding its applicability in a smart home. In the final chapter, a conclusion is drawn and a short outlook on future extensions is given.

1.1 Building Automation

Building Automation (BA) is concerned with the control of various devices found in the built environment. It aims at the improvement of control and management of mechanical and electrical systems in buildings [6]. Traditionally, the core domains of BA were limited to Heating, Ventilation, and Air-conditioning (HVAC) and lighting/shading. There, dedicated control systems known as Building Automation Systems (BAS) took (and today still take) over the automatic control of a range of dedicated devices. These devices can be categorized into three main types: sensors that perceive the environment and collect information on it (e.g., measurements), actuators that can interact with the physical environment and change its state (e.g.,

mechanically) as well as controllers that collect information from sensors, compute set points for the process and control the actuators accordingly.

Historically, the main motivation for the use of BAS is found in the exploitation of previously unused savings potential. With the help of BAS, a significant reduction of a building's energy consumption and thus a reduction of operational costs can be achieved. Closed and open loop control of environmental parameters or device functions are taken over and executed automatically by the BAS. Apart from the (energy) optimized control of building functions, additionally also the availability of a central management interface to all devices, the possibility of pervasive data collection and the easier, centralized re-configuration of control zones and parameters were drivers for BAS installation. Furthermore, BAS enable advanced functions, such as remote access to data and devices, the detection and forwarding of alarm conditions and alerting. However, the deployment of BAS was – and partly still is – characterized by high initial investment costs and thus a long Return on Investment (RoI) [7]. For this reason, building automation was mainly considered for functional buildings such a airports, hospitals or office buildings until some years ago.

Figure 1.3 illustrates the core domains of building automation. Apart from HVAC and electrical installations, today also the domains of security (e.g., access control systems) and safety (e.g., fire alarm systems) are integrated in the building automation system. In each domain, specialized devices assure the interaction and control of the associated physical processes, for example of temperature control in the HVAC domain. The interconnection of the different devices is accomplished with the help of so-called *control networks*. These are networks of different dedicated technologies that are tailored to the exchange of process control data and assure robust data transmission with the (mostly relaxed) timing boundaries of the applications. On top of the automation domains, supervisory control of the whole system is offered by a *Building Management System (BMS)* or by a *Supervisory Control and Data Acquisition (SCADA)* system. These systems offer management functionality of the whole installation to the operators or remote servers, and have a global view of the underlying systems.

BAS were initially designed and mostly realized following a *three tiered architecture*, where *field*, *automation* and *management* level were differentiated (cf. Figure 1.4). This classification according to device functionalities includes that at the field level, sensor devices collect information from the environment or the process that is controlled. Actuator devices are responsible for controlling and influencing the process by directly interacting with it. The interpretation of sensor values and global

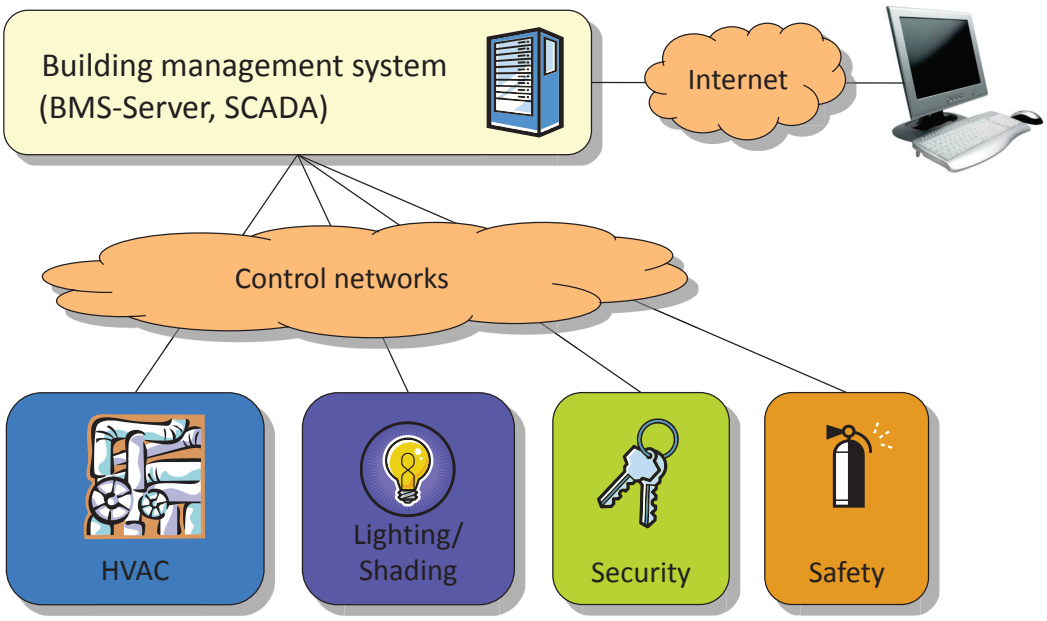

Figure 1.3: Domains of building automation

control parameters as well as the calculation of the desired output values and settings for the actuators are achieved at the automation level. There, automatic control, such as open or closed-loop functionality is realized by dedicated control devices, such as a Direct Digital Controller (DDC) or Programmable Logic Controller (PLC). In the management tier, management and configuration tasks are realized. Global functions such as visualizations as well as domain- or building-wide actions (e.g., a central lights-off functionality) are implemented here. The advantage of such a hierarchical structure as used in BAS and the resulting distributed control is its robustness against device failures. Despite of single devices becoming inoperable, this error is normally not propagated to the whole system but most control and management functionality remains fully available.

However, the classical view of BAS is no longer of practical importance. With the ever increasing processing power of computers and mainly also the frequently used embedded systems, the original tasks of the automation level were split up and re-assigned to both remaining field and management tier. This led to a more functional view in BAS compared to the previous highly device-centric model. In the resulting *two tiered architecture*, smart sensors/actuators themselves take over parts of the control functionality, and mostly PC-based workstations are used to

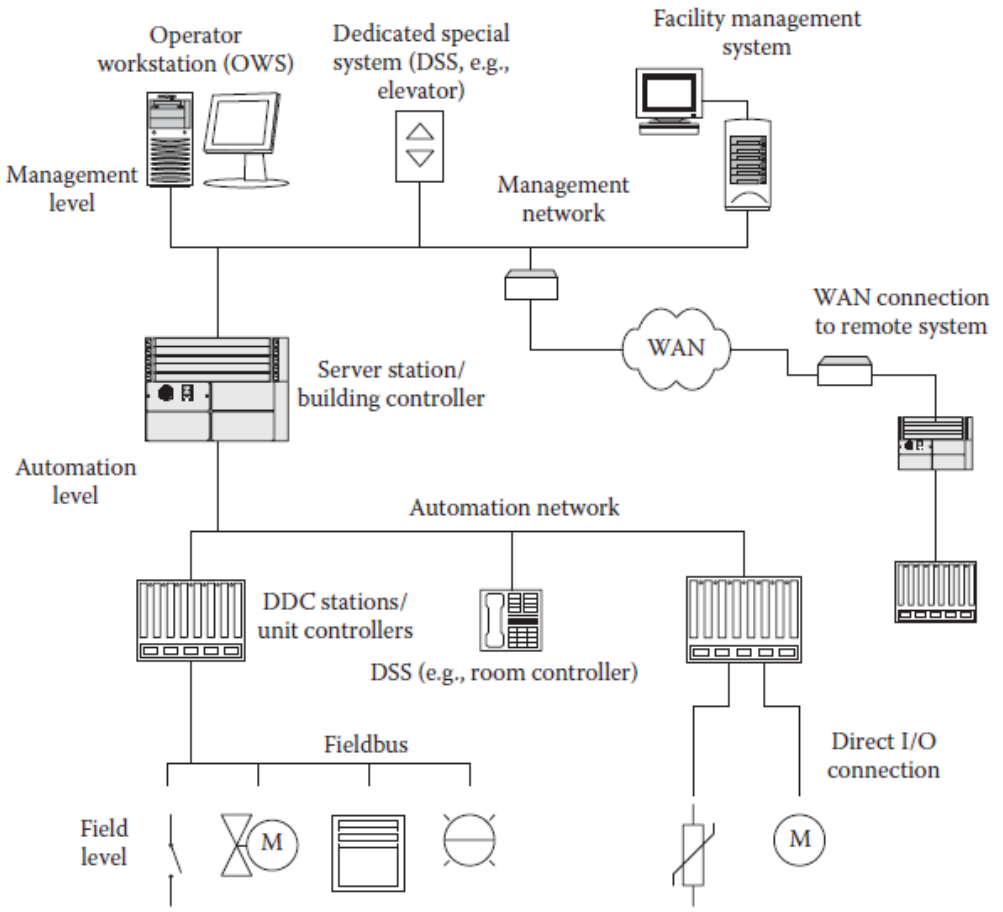

Figure 1.4: Three tier architecture of a building automation system with field, automation and management level and respective devices [8]

execute higher level automation as well as management tasks. Apart from the new distribution of device functionalities, the transition towards two layers also implied a change of the network structure. Practically, modern BAS consist of a number of *field network* segments that are interconnected with the help of a common *backbone*. This integration approach [9, 10] today almost always results in the traditional fieldbusses being interconnected over an IP network segment that is often also shared with non-automation related data transfer, such as for office communication. The resulting network architecture is shown in Figure 1.5.

The field networks enable the data exchange that is necessary for the operation of the control tasks and strategies that are realized in a decentralized way. Thus, not only sensors and actuators, but also automation level devices, such as Direct

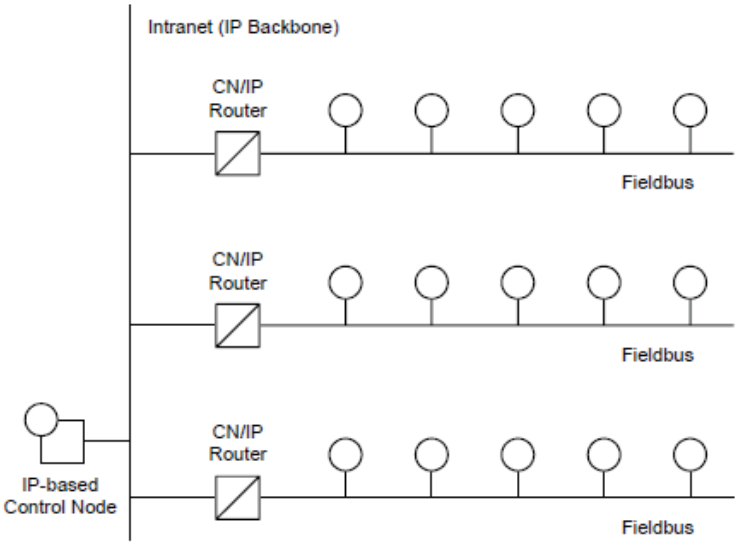

Figure 1.5: Two tier architecture [11]

Digital Controllers (DDCs) or other controllers are used there. One advantage of this architecture is that it allows for a flexible extension of the BAS. By adding additional field networks, the system can be extended in terms of devices, thereby for example adding control for another room or building wing. The extension can also concern the amendment of further functionalities to the BAS, where a new field network can add support for new control possibilities, such as access control. It is common that single field networks use only one specific BAS protocol and that their networking technologies are highly tailored to BA specific domain requirements (low data rate, relaxed timing constraints, robustness). In other words, the network and application layer protocols are commonly not mixed within one field network segment[2]. The separation into field networks introduces the benefit that different networking technologies can be used in the overall BAS. It becomes possible to mix wired and wireless networks, or more generally, networks and protocols with different characteristics, and hence to choose the best-suited alternative for specific automation tasks. Likewise, this also implies that Quality of Service (QoS) parameters, such as secure communication or communication latency can be guaranteed more easily by simply choosing suited protocols. Today, the separation line of field networks thus runs not only along system technologies but equally along application

[2]Although this would be theoretically possible because many established protocols rely on similar physical and data link layers, e.g., Twisted Pair (TP).

areas. In particular, several application specific protocols have been developed in the last decade (cf. Section 1.2.2) that are tailored to take over specific tasks, such as metering or advanced lighting control and that fulfill specific QoS requirements. These are also realized as single field networks.

The resulting mix of heterogeneous field networks is interconnected using a dedicated network segment called backbone. This is for several reasons. First of all, the backbone is home for management level devices. These are used, on one hand, as central configuration points to define global control parameters (e.g., setpoints) and also to initiate global control tasks (e.g., emergency shutdown). On the other hand, a global view of the whole BAS is desired. It is provided on the backbone level, where all data from the field networks is – in principle – available or can be accessed. Therefore, current backbone networks are mostly Ethernet and IP based. Visualization, trending and alarming functions hence use the aggregated view of the underlying devices and data points. In practice, the backbone network fulfills another important task: it achieves a physical interconnection between different systems, which is a prerequisite for the realization of multi-protocol interactions. In practice, however, the transparent exchange of process data, such as sensor values across different protocols is of interest. In this case the simple physical interconnection of network segments via a common backbone (OSI network layer) is not sufficient to enable true interaction among the different protocols. For a full integration, the translation of communication specifics **and** semantics (i.e., up to OSI application layer) across protocols is needed. This integration of heterogenous systems and protocols has been one of the main challenges in BA for the last decade. In course of the evolvement of Smart Homes this topic has again gained considerable importance.

1.2 Smart Homes

The term Smart Home (SH) generally denotes a residential dwelling where automation technology takes over the automatic control of (at least) parts of the building services. It shares its roots with BA, but nowadays needs to be regarded as an independent application and research area. In order to differentiate it from "classic" BA, it is often referred to as Home Automation (HA). In practice, systems are often developed for the use in both HA and BA. For this reason, the term Home and Building Automation (HBA), reflecting the common ancestors, is frequently used to refer to automation systems that are used in buildings of all size and all different usage types. Another related term is *domotics*, which explicitly relates the Latin

term "house" with one or more other words from the set of "informatics", "robotics" or "automatic". Like HA, it therefore refers to the use of automation systems or robots in a home environment. Over the years, domotics has been superseded by the more generic terms of HA or SH. However, domotics is nowadays still used quite frequently to describe the use of robotics in a home and to refer to the *Ambient Assisted Living (AAL)* domain [12].

Figure 1.6 illustrates the different domains of a smart home. Apart from the "classical" automation domains HVAC and lighting/shading, today also systems that ensure security and safety in the home are covered by automation systems. Additionally, a smart home also interconnects household appliances as well as consumer electronics and entertainment devices. Moreover, equipment from the AAL domain (e.g., sensors monitoring human vital parameters) are integrated in a smart home system. While in building automation only dedicated automation networks enable data exchange, the consideration of further domains and application fields in the smart home introduces also new networking technologies in the home. These are mainly IP based in order to ensure the transfer of larger data quantities and hence have considerably different characteristics than the traditional control networks. Also in smart homes, a centralized access point to the system is desired. This smart home server can be implemented, for example, on a set-top box shared with the TV/entertainment system. The home server, on one hand, provides a central configuration and management point of the underlying systems and devices and may offer for example a visualization and services for trending/logging. On the other hand, this central instance acts as a gateway to and from the outside world, in particular the Internet and Internet-based applications (resting upon e.g., Web services).

Finally, today some vagueness concerning the terminology exists in literature and research. The terms HA and SH are often used quite synonymously although they indicate some different approach concerning the application of automation systems in the residential home. For the sake of clarity, there shall be made a clear distinction between them in this dissertation. The older term of HA shall be used to express the classic, state-of-the-art approach of using automation systems in a building. It always implies that automation technology is exclusively used to control the core domains of a building (i.e., HVAC and lighting/shading) and shall express that only standard control approaches are applied (e.g., established PID control loops). In contrast to this limiting definition, the term Smart Home shall be used in this dissertation to explicitly point out the use of automation systems in further areas, thereby integrating additional building services and devices, such as entertainment

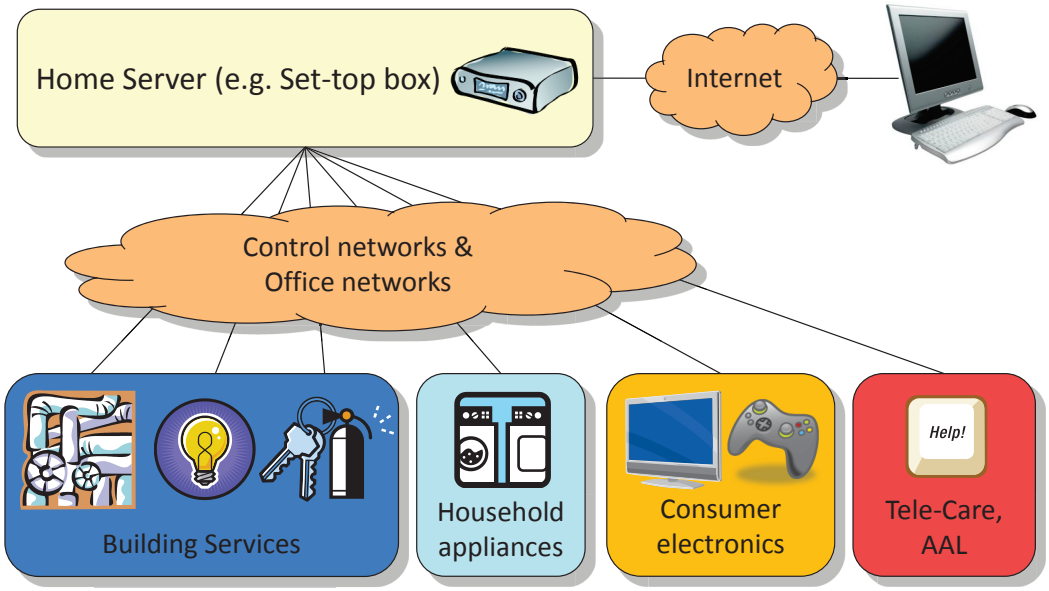

Figure 1.6: Domains of smart homes

devices and household appliances. The "smartness" of the smart home evolves from the mandatory capability of a smart home to reach its control goals in an intelligent way. This means that a smart home has to be capable of selecting the most adequate[3] decision among a number of different control possibilities. In the smart home, this comprehensive control of the building services is realized by advanced control strategies. The control strategies are implemented as algorithms operating on top of the automation system. Their smartness is mainly made up of and can be described as intelligent decision making under consideration of all available information on the current and probable future world/environment states. As control strategies that realize such an intelligent behavior can be highly diverse, it is necessary to have a closer look on the basic elements that induce intelligence in the context of smart homes. Some requirements that especially characterize smart homes and differentiate them from "standard" home automation systems are summarized in the following list.

- Information aggregation and analysis: The collection and analysis of distributed information is the most basic indicator for a smart system. Information can be gathered from devices over the underlying automation networks as well

[3] *Adequate* can be interpreted differently according to the inhabitants' preferences. It may refer to the most energy-efficient, comfort preserving or other control choices.

as from additional auxiliary data sources, such as the Internet or other data repositories. In particular, some (rudimentary) intelligent behavior can be derived from linking and joint analysis of the data. One very basic example is the consideration of weather information within the room temperature control algorithm.

- Context awareness: As another basic form of smartness, context awareness is closely related to the before mentioned availability of information. The information is analyzed and related parts are set in context to each other. This way, specific information patterns can be identified as situations, while other information represents the context of this situation. Thus, it becomes possible (for computers) to automatically identify, for example, a set of specific user actions or a set of environment parameters, and group this information as situation [13].

- Parameter prediction: Prediction capabilities enable a control system to make better decisions thanks to additional information on future events being available. In a smart home setting, control strategies require specific parameters as input. For example, room temperature control algorithms take some setpoint temperature as input which the system subsequently attempts to reach. If predictions on such parameters are known, predictive control strategies can be applied. Following the temperature control example, this enables to start heating/cooling processes at the most suited (e.g. most energy efficient or latest possible) point in time. Similar, predictive control can also be applied to other building services, e.g., lighting/shading [14].

- Action/Event prediction: Similar to the prediction of parameters, intelligent control can also benefit from information on future events in the home or environment. Typical examples for such events are changes in occupancy or of the weather situation. Actions, in this context, denote activities that are initiated or executed by users. For intelligent control, knowledge about regular activities, e.g., switching on the TV at regular times can be incorporated and used to improve control strategies.

- Learning capability: An important capability of a smart home is its adaptability to situations and users. In other words, intelligence in the home is also characterized by the smart home being able to learn from and adapt itself to new or changed situations and parameters. Seen from a user's perspective,

a learning capability for example allows to keep track of a user's preference and translate or map user actions into changes of the considered parameter. To yield full benefit, learning capabilities can thus be combined with context awareness mechanisms, where additional classification possibilities arise.

- Pro-active actions: A smart home can also be characterized by its autonomous ability to interact with its environment in a proactive way. Compared to traditional home and building automation systems that follow a stimuli-reaction process where actions are initiated only as reactions to some user or sensor input (closed loop control), intelligent systems may execute actions at any time and especially without requiring any prior stimulus or event. The trigger for the execution of some operation may thus be a timer or the result of an internal deliberation process.

Similar to the above requirements which were initially derived from the automation system perspective, also Aldrich identified several criteria that characterize a smart home [15]:

- Homes which contain intelligent objects: A home fulfilling this class has several stand-alone appliances and objects that are equipped and react with some sort of intelligence.
- Homes which contain intelligent, communicating objects: In order to add and improve functionality the appliances can exchange information between one another.
- Connected homes: The home is equipped with a network allowing interactive and remote control of systems. Also services and information can be accessed from within and outside the house.
- Learning homes: Activities and patterns in the home are recorded and stored to anticipate the needs of the user and also control the technology accordingly.
- Attentive homes: The home is aware of activities and locations of the users as well as the objects. With this information the house reacts to and predicts the user's needs and controls the technology accordingly.

It becomes obvious that Aldrich's smart home categories overlap in most parts with the previously mentioned ones. However, the latter definitions are in one respect more generic (e.g., objects versus concrete devices), and in another difference demand

location information on devices and users to be available. This "attentiveness" extends the awareness criterium from above especially with respect to the location of objects. In this dissertation, an exact location information is not considered critical to control capabilities. Rather, knowledge on device and user locations in the granularity of rooms is deemed sufficient for the targeted control tasks.

A comprehensive definition of a smart home based on all the criteria explained above is summarized as follows:

Definition: A *Smart Home* is a dwelling in its residential meaning, i.e., a house or apartment, in which automation technology is used to control at least two different building services. The operation of a smart home is generally computed and executed autonomously and automatically by dedicated control algorithms and programs. Manual interventions remain possible at any point in time and are treated with priority if they are not inflicting hazardous or safety critical situations.

Smart homes always operate under the regime of energy efficiency and the concurrent requirement of maximized inhabitant comfort. Key criterium is the global optimization of comfort and energy efficiency on a dwelling level. For this optimization task, information technology is exploited to host and execute control algorithms that are capable of multi-criteria optimizations. Minimum input for these algorithms are goals of a global level, the states of the automation system and the devices as well as information on the state of the environment. This mandatory context awareness as well as the optional availability of predictions on future environment states (anticipated user presence, predicted weather situations) are further foundations of smartness.

The following graphics provide an overview of common smart home services (cf. Figure 1.7) and illustrate typical devices and according services that are already realized in the smart home today (cf. Figure 1.8).

Home owners have different motives to use automation technology in the residential area. Compared to BA, the area of SH is a relatively new topic and the public interest in it is still growing. The increasing importance can be amounted to a considerable extent to the constantly increasing prices for energy [4] as well as the simultaneously tightened legislative regulations for pollutant emissions, such as greenhouse gases. Also the increased building of low- or even zero-energy houses contributes to the further spreading of smart homes. For low-energy houses, the use

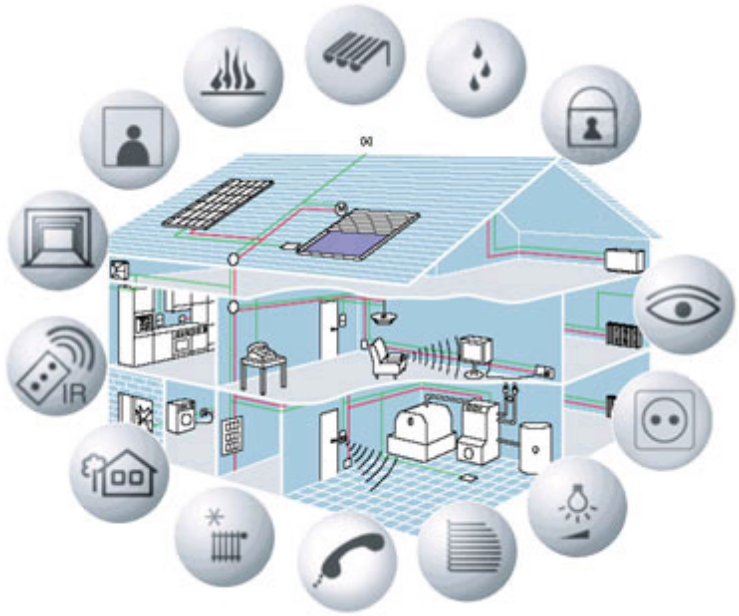

Figure 1.7: Overview of smart home services [16]

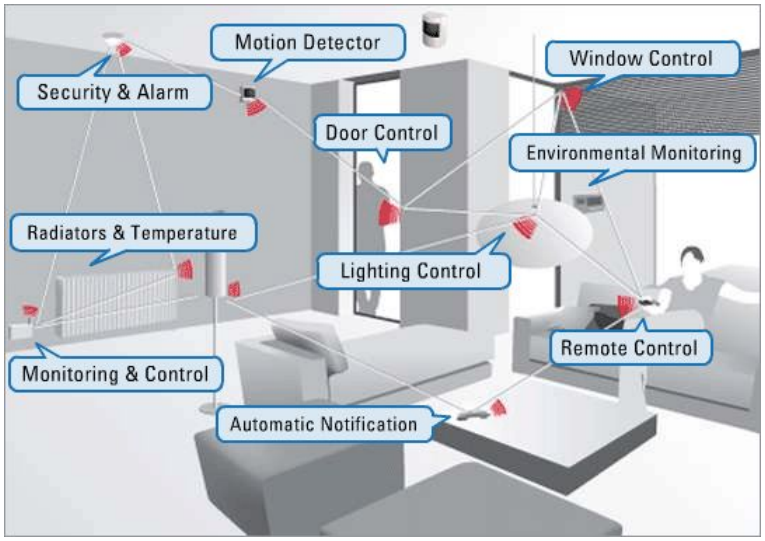

Figure 1.8: Networked devices in a smart home [17]

of automation technology to control heating/cooling and airing systems is a mandatory prerequisite. Therefore an extension of the already existing underlying system

towards a smart home introduces less financial and technological overhead compared to completely new installations or later refurbishment.

Another current development that is strongly connected to smart homes is decentralized energy production including the integration of the end users in the so-called smart grid. In order to use self-produced energy efficiently, control mechanisms are needed that can deliberate alternatives and automatically execute selected tasks in the home. Again, the instantiation of these requirements can be found in the automation systems deployed in the SH. In this context, also the ambiguous term *plus-energy house*, where locally produced excess energy is re-fed in the electric grid, needs to be mentioned. Similarly to zero-energy buildings, control algorithms are needed to detect and deal with an excess energy production. It then depends on the implemented control strategies if and how much energy is used locally or re-fed into the electric grid to yield monetary benefits. Hence, viewed on a greater scale, each single SH is also a self-contained unit in the electric grid. This view enables higher control logic to locally distribute energy on an area/district level, where the SH are regarded as black boxes with (only) energy input and output.

1.2.1 Smart Homes in Context

A smart home has several boundaries to other (research and other) disciplines, systems and persons. In general, knowledge of the context of a smart home and knowledge of prospective users contribute to a better understanding of smart homes and facilitate a later system design and implementation. The goals of external systems as well as the aims of the different users and other involved persons make up and illustrate also some further expectations that are put on a smart home.

Among these overlapping points of smart homes and the environment they are set in, the group of smart home stakeholders is most important. In [18], stakeholders are defined as all persons that have an interest in a particular project. In case of the smart home, the definition of stakeholders is slightly extended and used to identify all persons and moreover all systems that act as agents in the smart home. Furthermore, some selected research disciplines that have a great influence on the concept of smart homes will be discussed separately.

Smart Home Stakeholders

In this section, an overview of all stakeholders that are involved in and around a smart home is given. Their relevance to the system and possible interaction points

are elaborated. The purpose is, on one hand, to identify all influences to the system and to consider the stakeholders' interests in the best possible way in the system design. On the other hand, a complete lists of stakeholders also facilitates the clear definition of the smart home system borders.

- Home owners/Permanent residents

  Without question, the inhabitants of the smart home are the most important group of stakeholders. Any smart home system that is installed in a home will first and foremost have to support the permanent residents of the dwelling, which are also the primary users of the system. In particular, the users have different expectations towards the system, which often correspond with the initial motivations that led to an installation of an automation system in the first place. Typical expectations are increased comfort, cost reduction, ease of use, peace of mind in daily operation, safety, security, or prestige. Since home owners are not necessarily technically versed people, they often prefer a system that is usable for them and that takes over routine tasks from them. Also, permanent residents spend much time in the home and hence might expect a system that learns their preferences and is able to adapt to them.

- Guests/Non-permanent residents

  This group of stakeholders also influences the system, although these people do not live permanently in the home. Nevertheless, they expect from a smart home to cater for their preferences, i.e., provide them with comfortable conditions and also support them in their daily tasks if applicable. The differences to permanent residents are that the guests' preferences are known only seldom (in case of recurring visits), that their presence and preferences are hard to perceive from the environment since their presences are often short compared to residents that live permanently in the home. Furthermore, it cannot be expected that guests are at all familiar with the general concept of a smart home.

- Cleaning staff

  Cleaning staff is a special kind of non-permanent inhabitant of a smart home. For one aspect, cleaning staff may not be familiar with the concept of smart homes. Furthermore, cleaning personnel may take unconventional actions within the home, which are in relation with their tasks (i.e., cleaning) but which may contradict or interfere with the smart home control strategies (e.g.,

the action of opening windows so that the floor can dry faster interferes with a strategy to keep a certain temperature level in a room). Cleaning staff therefore introduces the requirement that the system must provide tailored modes that can be activated in special situations.

- Government/Society

  Although not directly involved in the daily operation of a smart home, the government must nevertheless be considered as one influential stakeholder. This is due to the extensive amount of regulations and laws that directly and indirectly impose limitations on the way a smart home system can operate. On one hand, laws concerning building architecture as well as building materials and structure greatly influence the final design and construction of the building. The building structure is the envelope wherein the smart home system must operate, so that these laws have implications on the system design. On the other hand, many countries have laws in effect that regulate the allowed emissions of buildings and associated systems, such as heating systems. Additionally, general emission limits and emission reduction targets may be in place (e.g., CO_2 emission reduction or the 20-20-20 EU targets) that influence also the smart home, for example by demanding the integration of renewable energy sources. Finally, laws and regulations are frequently found concerning the protected operation of buildings and services (safety). Also these regulatory issues must be covered by the smart home. Similar requirements are imposed on the smart home by the society. Although the pressure exerted by the public (e.g., the public demand for emission reductions) is not legally binding, the public opinion is a major factor in any larger scale building project. Thus, such demands from the society often need to be reflected in the design and operation of a smart home.

- Energy providers

  Especially when an integration into the smart grid is in question, energy providers represent further stakeholders. Applications, such as demand side management that require the energy provider to take over some control of selected building functions, services or devices may be executed in a smart home. The system must provide dedicated interfaces so that access to the desired building services becomes possible. However, this access must be configurable, limitable and revokable (access rights policy).

- Facility managers

  Facility managers are concerned with the daily operation of a building. Although their services are predominantly used in functional buildings today, future smart homes may also make use of their offers. This may concern applications, such as the control of building functions, monitoring of parameters and the analysis of captured data, for example to later use it for differential cost attribution. Out of these reasons, facility managers have to interact with a smart home and are thus also stakeholders. Their main interest in the system is the availability of (aggregated and technology-independent) data and respective interfaces and services to access this data.

- Miscellaneous service providers

  In the smart home, miscellaneous external systems may connect to the smart home in order to perform some tasks. These are, for example, information providers, such as weather report services but also information consumers that might also want to actively execute some tasks in the smart home, such as security providers, rescue and alarm service providers or performance contractors. Since all of them have to interact with the smart home, they are automatically stakeholders of the smart home system. For them, the system must provide adequate interfaces for interaction but also tackle security and privacy issues of the data within the smart home, e.g., personal data.

- System engineers

  System engineers are responsible for the installation, configuration and maintenance of a smart home system. Their interest towards the system is to be offered with an interface that allows a deep access into the system. In the best case, these technicians rely on remote system access to execute configuration and maintenance tasks. System engineers furthermore require access possibilities to very technical aspects and normally hidden system parameters and functions. They hence influence the system design in a way that different interfaces and access points for different stakeholders must be designed and offered during the smart home operation.

- BAS equipment manufacturers

  BAS equipment manufacturers are representatives of passive stakeholders in the smart home. Although they hardly interact with the system at runtime, they provide the basic building blocks of the smart home, namely the devices and

according systems that interconnect them. In this respect, the kind of devices and functionalities that are provided by them as well as the way their systems can be programmed and maintained has an influence on the system concept of a smart home. In particular, a smart home system must be designed to operate on the foundation of its device landscape and thus needs to consider the related requirements in the smart home concept. Additionally, building functionalities that are not yet offered by BAS devices cannot be trivially integrated in the smart home[4].

Smart Home Research Disciplines

Apart from the group of stakeholders that all have a direct system interaction in common and are hence essential to be considered in the (software) system design, the smart home is also characterized and influenced by other, mostly research disciplines. In contrast to the stakeholders, these disciplines are not involved in the daily system operation but nevertheless hold an important share of a successful realization of a smart home. In a timely context, the disciplines listed below take decision at an earlier point in time than the smart home system design. Thus these decisions may considerably influence the smart home and so knowledge of the related disciplines is of great importance.

- Architecture/planning

  Architects and planners are involved in the design and construction of a building from the very beginning. With their design and the technical concept of the building, they influence the later possibilities that can be offered by a smart home. Under their influence, parameters, such as the orientation of the building, its window and door openings, but also decisions of the used building materials and of course the general architecture are decided. Concerning the smart home this influences the control efforts (e.g., cooling is more complex and energy-consuming if many glass fronts are present) but also practical issues, such as the placement of devices which might be limited due to aesthetical and practical reasons (e.g., sensor placement is not easily possible on windows). Furthermore, planners often act in close cooperation with the architects and define the availability of building services in the home. This includes decisions

[4]Nevertheless, a novel smart home can and should envision the use of future technologies, functions and devices. Only then, it will be a future-proof solution and provide progress beyond the state-of-the-art. This justifies, among others, the integration of technologies and functions that are not yet ready for the market due to costs, complexity or similar criteria.

on the heating/cooling systems, lighting/shading equipment but also on the installation of renewable energy suppliers (e.g., geothermics).

- Building physics, building science

  From the research part, the disciplines of building physics and building science play an important role concerning smart homes. Building science researches thermal and visual (with respect to lighting/shading) aspects of a building, integrates further aspects, such as ambient energy, acoustics, humidity and also is concerned with the operation of the building/smart home. Building physics is also the theoretical basis for the aspect listed before in a way that research there provides the theoretical backgrounds for the physical processes that happen in a building (e.g., the energy demand for heating of a room under consideration of parameters, such as building structure, thermal conductivities of building materials, heat gains from equipment and many more). Hence, building science influences the possibility and effectiveness of the different control strategies and can deliver important inputs for the energy efficient operation of a building.

- Control engineering

  The control strategies are the subject of research in control engineering. Control engineering, as the name implies, is concerned with research on and design of the control strategies that are later implemented and executed in the smart home. Control engineering hence aggregates selected aspects of many of the disciplines and stakeholders and realizes algorithms under consideration of these building blocks. In a smart home, an increased variety and amount of these building blocks compared to "standard" homes is offered per definition. Thus, the consideration of control engineering expertise has to be considered especially fruitful there.

1.2.2 Technologies for Smart Homes

Over the last three decades, many different standards and protocols for the use in building or home automation or both of them have emerged. These are also the basic building blocks of today's smart home systems. In a smart home ideally an all-in-one solution that allows total control of all conceivable scenarios in the home would be desired. However, even despite the long timespan of their development, not one specific protocol has yet emerged that covers all application domains of interest in a smart home. Rather, many different protocols co-exist. Some of them aim at

the control of multiple domains, while others exclusively offer tailored functions for a specific area of use. These protocols have in common that they were developed as dedicated communication protocols for home or building automation systems. Therefore, all protocols are tailored to the specific requirements of this domain. They are robust against data loss, can accommodate a high node count, and focus on the transmission of building control data which is little in size. Especially the latter requirement influences the design in such a way that protocols offer only limited throughput and feature a rather high latency. This means that mostly no real-time capabilities are offered because the transmission performance is not considered critical[5]. Furthermore, many protocols reflect the demand for a low cost per node and can therefore be implemented on hardware platforms that offer only limited resources, such as computational power or memory.

Among the plethora of smart home protocols, three major standards can be identified that cover multiple domains of the smart home and whose standards are furthermore openly accessible[6]. These are BACnet [19], KNX [20] and LonWorks [21]. The open standards have in common that they are application-independent and can be used at all three levels of automation systems. Although all of them provide functionalities for more than one application domain, they are not completely versatile. In particular, BACnet, KNX and LonWorks all cover the traditional core smart home domains HVAC and lighting/shading, but lack detailed support for home entertainment equipment. In contrast to these open standards, a continuously increasing number of proprietary standards exist. Their main drawback is that protocol details are not disclosed, which makes them not suitable for integrated smart home systems that have to cover a broad range of different functionalities. Additionally, such proprietary systems cannot be thoroughly evaluated regarding their, for example, security features. Their devices are usually only available from one manufacturer, which induces the risk of a vendor lock-in situation. Thus, in this dissertation, the focus is put on the use of the aforementioned major open standards, because they provide most of the main important functionalities of a smart home and can be analyzed in depth thanks to their openness.

[5]The only notable examples of time critical data transmissions in smart homes concern lighting and security/safety services. However, both of them can be satisfied with reaction times of some hundreds of microseconds, which is regarded as a relaxed time constraint when compared to industrial automation scenarios. Additionally, security and safety applications rather demand transmission reliability than transmission speed.

[6]"Open" in this case refers to the general accessibility of most or all protocol details. Nevertheless, also standards that are denoted as openly available may require a membership to some institutional body (e.g., the KNX association) and/or the prior payment of access fees.

Apart from these versatile open protocols, also solutions exist that are dedicated to particular application domains. These protocols are characterized by their support of application-specific functionalities and can be differentiated according to the commands, communication types, or miscellaneous other features (e.g., security) they support. One particular class of them are wireless standards, such as ZigBee [22], EnOcean [23] and RFID [24], which provide cable-free communication services but also come along with drawbacks, such as very limited power availability and computation resources as well as proneness to both malicious and unwanted interference. Furthermore, standards and communication protocols exist that provide tailored services for a particular application scenario. For versatile lighting control the DALI standard [25] is available, the SMI (Standard Motor Interface) bus [26] is mainly used for shutters, protections systems and sunblinds, entertainment devices can be controlled with the help of DLNA [27] , while metering services for electricity, water and other physical values are provided by M-Bus [28].

1.2.3 Challenges for Smart Homes

Although automation technologies have been available for several decades and have evolved greatly over the past 20 years, smart homes are still not used pervasively today [29]. Several reasons for the rather reluctant use of automation systems in the home compared to their frequent deployment in the industrial domain can be identified. Some of them are rooted in shortcomings of the technological basis (e.g., integration, application variety, autonomy), while others reflect personal reservations of the targeted user group (e.g., usability, economic considerations). The main issues that currently limit a broader spreading of automation systems in the home market are listed below.

- Integration of application domains

  Especially in a smart home, automatic control can no longer stay limited to the classical automation domains of HVAC and lighting/shading. Rather, the control possibilities need to extend also towards the increasing number of electric consumers in the houses, such as larger household appliances and entertainment devices. To maximize the benefit for the home owner, a smart home needs to integrate all these different devices and provide control possibilities for them. However, this is not a trivial task due to several reasons. First of all, already the core domains of automation historically evolved separately. This led to a large number of dedicated systems that cater well for (domain-) spe-

cific use cases but often do not foresee any extension towards other domains (cf. Section 1.2.2). With the only recently increased availability of networked home electronics and appliances, (automation) history repeated itself. Also in these new domains, one universal communication standard is missing, and not one particular universal winner candidate can be identified yet. The situation is worsened by the fact that currently it may happen that not even entertainment devices that are manufactured by different companies can communicate with each other by default. This shortcoming can, in many cases, not only be attributed to technical necessities but is often a product of manufacturers' interests.

Regarded from a smart home viewpoint, the current situation demands that mechanisms (such as protocol adaptations, gateways, integration applications) need to be developed that solve the integration problem at least for the smart home. Still, the ultimate goal would be the availability of one universal smart home standard which connects to all smart home devices and provides tailored application-level functions for their access and control.

- Representation of a homogeneous system/central access and configuration point

  Related to the required integration of a highly heterogeneous device landscape and the fragmented protocol situation connected to it, smart homes need not only achieve an interconnection of the different devices, protocols and application domains. Even more important is the collection or aggregation of all devices and systems, followed by their representation in a unified way. Only such a protocol- and application domain agnostic representation guarantees that novel applications that connect different application domains can be implemented. One benefit that comes along with the integration is the introduction of a centralized access point to the overall system. It provides users and applications with a centralized view of the (aggregated) system and can thus be used for all global tasks, such as configuration, visualization or logging and trending. Additionally, maintenance personnel or system integrators need no longer have a detailed understanding of all different systems but can operate on top of the aggregated system. This facilitates the engineering and configuration processes and hence reduces time effort and costs of the engineering specialists. The same is the case for later re-configuration and maintenance tasks.

However, the aggregation towards a homogeneous system introduces several challenges. The easiest solution would be to define a new protocol standard, which comprehensively caters for all application domains. It would furthermore have to be widely accepted by industry as well as consumers and provide flawless communication for all heterogeneous devices. Unfortunately, such a world standard is not easily designed and much less it would be accepted by industry. Thus, the only solution that remains possible is the re-use of existing, established and open standards that are combined at some higher layer.

The main drawback with this solution is the required translation of protocol specifics into a more generic, universal system. On one hand this translation makes scenarios available that combine devices of different application domains and protocols. It further facilitates the integration task because now *generic* functionalities are bound together to applications, and thus relieves the system integrator from having to know system internals.
On the other hand, a generalization and service translation always comes at the price of information loss. In order to have generic commands to access building services, such as "*light on/off*", it is necessary to map all protocol specific commands to these services and to translate them when accessing some device, respectively. Unfortunately, this translation is not always straight forward and may be accompanied with the loss of some protocol features. For example, if protocol A offers a command "*dim light %*" but protocol B only supports "*light on/off*", the most common integration choice is to support only the latter command at the generic level, thereby following some kind of "least common denominator" principle for protocol services. The reason for this is that the generic representation must feature only commands that can be executed by all underlying systems. Another option would be to transparently offer all services, regardless if they are generic or not, to the application designer. Concerning the example given above, this would result in commands in the fashion of "*protocolA_dim light*" and "*protocolB_light on/off*" and hence in the drawback that again in depth protocol knowledge would be required to know which commands are compatible with each other and with which devices.

Given the problems and challenges discussed in *Integration of application domains*, it is important to note that the common representation is the benefit of a logical overlay put on top of any underlying technical realities. When looking deeper into the smart home system from bottom to top, it will remain obvious that the overall system is de facto fragmented into several, more or

less independent subsystems and into the different devices that are connected there. However, a unified, homogeneous view is presented to the applications realized in the smart home as well as to its users. All participants can (and should) access the system and its devices in a top-down fashion and through a centralized access point. Nevertheless, also the direct access on a specific protocol level remains possible. However, such a direct access on a lower system level is no longer advisable, as it may influence higher level (domain-spanning) applications. If such unwanted dependencies remain unrecognized, even total system failures may result. Thus, once a generic way to access a system built of multiple heterogeneous subsystems exists, the manipulation possibilities at a subsystem level are severely constricted. It must therefore be the goal for all integration efforts to keep the translation losses at a minimum, to preserve protocol specifics whenever possible and nevertheless to provide a central access point with a set of generic services.

- Retrofitting

  Buildings in general and with them systems that are deployed in the home are characterized by their long lifespan. In contrast to consumer electronics and sometimes also home appliances which are typically used for several years only and exchanged afterwards, the typical usage period of a house has to be measured in human generations. Refurbishments of a building hence mostly occur only every several decades.

  This has two major implications for smart homes. First, automation systems that are installed in a home need to be a future-proof solution that is able to fulfil the requirements that may arise in the future. Second, smart homes should not only be realizable in newly built environments but their concept should also be applicable to the huge number of existing older buildings[7]. The deployment of smart home technology in existing buildings is even more important because the total optimization potential in absolute values (achievable by, for example, using advanced control strategies) is often higher in these buildings than in newly constructed ones. Therefore, retrofitting of smart home systems must be possible. This concerns networking technologies on one hand (where for example wireless networks or powerline communication can be used), and demands flexibility of the system on the other hand (cf. *Adaptability*). In this

[7] Older in this case not necessarily refers to the age of the building but also to the age of the available equipment, building components, such as doors and windows, the availability of modern networking technology, etc.

context it is important to also demand that smart home applications should neither be operable on the newest equipment only nor only in case of completely automated buildings but there need to be solutions for partly equipped houses, too.

- Adaptability and modularity

  Today, it is still often the case that automation systems are installed and configured once at the time of their first deployment, but that these initial configurations are hardly ever changed again. Furthermore, the case that an existing automation installation is extended with additional automation equipment is not well reflected in the systems.

  Similar to the arguments brought forward concerning retrofitting, it must be feasible to extend or modify a smart home system with respect to changed requirements. These changes include the addition of further automation technology (e.g., to control previously not covered building services) as well as simple changes in the use of rooms or functions. To fulfill the first demand it is necessary to have open systems that in the best case detect and integrate new devices or services (almost) automatically, reaching out to the direction of plug-and-play functionality. The second challenge is that the smart home must thus have capabilities that allow to easily enter these changes into the system. New usage types must be recognized and considered in the control strategies, which requires the smart home to be able to reconsider or even adapt control strategies according to the available devices/building services and the planned use. Furthermore, the smart home should aim at maximized efficiency and comfort, thus adapting its control to cover also newly added domains/devices, which demands a modular system architecture that facilitates this process. If the requirements are not fulfilled, the smart home will have only sub-optimal performance and in the worst case even more energy (and thus more money) could be used than would be required by manual control and the dissatisfaction level of the user concerning the system performance may be high.

- Integration of the user: preferences and usability

  Smart home systems are built to support the energy efficient operation of building functions. However, this optimization has to happen under the regime of the residents' preferences, mainly their expectations concerning the desired comfort level. Therefore, any smart home system has to provide two further

characteristics: First of all, the system must know its users' preferences and include them in its control strategies. One of the most important success factors of a smart home is that the user always feels comfortable in his/her home. This often implies that the user is able to perceive the added value of the automatic control (i.e., the user comprehends the system to make good and reasonable control decisions), but nevertheless has the possibility to manually overrule the system at any time. The system should also learn from the users and thus provide personalized control (e.g., automatically recognizing that one window should never be opened). Furthermore, users should be confronted with a system that is capable of operating autonomously without requiring intensive interaction (i.e., routine tasks are executed automatically without the user triggering them). As a second main point, usability of the system must be guaranteed. If user inputs are required (e.g., during initial system configuration), the system must foresee suitable mechanisms/interfaces that can be used by the inhabitants to interact with the system. Hence, user interfaces should be intuitive and keep the cognitive load to a minimum. Additionally, the smart home system should be able to offer feedback in an appropriate way to the inhabitants. Such a feedback should be provided unobtrusively and be easily understandable and interpretable.

- Economic challenges

  Main drawbacks are the considerable costs that arise when a smart home system shall be installed in a building. Many costs can be attributed to the required initial investment in suitable (e.g., networked) devices, the accompanying network infrastructure that is connecting them and the considerable (personnel) efforts for installation and initial configuration of the system. Additionally, costs for later re-configurations and adaptations must be calculated, since these can in most cases – due to their complexity – only be accomplished by specially trained maintenance personnel. Another aspect to be considered is that with the integration of systems, also the overall system complexity increases. From an economic viewpoint this is reflected, among other points, in the required efforts for installation and (re-)configuration of smart home systems.

  From an economic viewpoint, any investment and therefore also the installation of a smart home system including a network, devices and control units has to provide a Return on Investment (ROI) within some time from the installa-

tion. In the building domain, a typical lifespan is denoted in the magnitude of generations, i.e. 20 to 30 years [6]. This long time period also imposes on the automation solution and the used devices to be future-proof, meaning that their mean lifetime has to be in the order of at least 20 years and that future system repairs and extensions stay possible (e.g., the system is still being available on the market).

However, especially in the home, automation systems are not only deployed out of economic reasons but also to achieve higher comfort and piece of mind as well as increasingly out of ecological considerations, which at least partly alleviates the economic requirements.

- Smart homes need to be recognized to advance the status quo of building service control

  One challenge for smart homes is that they still are not comprehensively perceived as a technological means to improve comfort and energy efficiency with superior possibilities compared to manual control. This, however, is important in several aspects. It justifies the considerable monetary investments into smart home systems, the perceived added value is required to further promote worldwide energy conservation and with it pave the way for buildings to become full members of a decentralized energy production and distribution grid.

  One main advancement that is not yet available is a system that supports the user in everyday tasks. The main idea of automation is to hand over tasks that are dangerous, boring or challenging for humans to machines that are able to handle them faster, more precisely or without getting tired. This idea is not pervasively realized in smart homes despite of the fact that home and building *automation* systems are available. It is also the case that several underlying functions in the building are related to complex physical processes. While these are only understood by domain experts (but not the typical home owner), computers have in any case superior computational power compared to humans trying to tackle the same problem. An example is temperature control, where a large number of influence factors, such as room size, persons in the room and their activities, environmental parameters, wall thickness, insulation, windows and their orientation and many more exist. Traditional control, even in an automated home, neglects most of them and thus may not achieve optimal results. Novel smart homes should address this shortcoming and thereby introduce a new way of building control.

In order to increase the percentage of homes that are equipped with automation technology and to promote a positive public attitude towards smart homes, it must be made clear that building functions can be operated better by a smart home system. This can be done for example by practical, publicly observable deployments of smart homes.

- Increase energy efficiency and comfort

  The main motives for the installation and usage of automation in the home are increased comfort and increased energy efficiency, the latter coming with the adjoint benefit of reduced operational costs. In reality, these goals of smart homes cannot be solved easily as, for example, the simple deployment of technology in the home is not sufficient. Additionally, unlike energy efficiency, comfort is a parameter that cannot be expressed in an absolute way. Rather, comfort is highly user-specific and also prone to change over time, but nevertheless most important to be fulfilled. In contrast to that, energy efficiency could be easily maximized if comfort remained unconsidered. Thus, a challenge connected in particular to both energy efficiency and comfort is that these goals need to be fulfilled simultaneously. A trade-off between them must be sought constantly during the operation of the smart home system.

  According to the definition given in this dissertation, smart homes furthermore demand that progress beyond the current state-of-the-art is made. Translated to this challenge, smart home owners expect better performance in terms of energy efficiency and comfort than the deployment of existing solutions would offer them today. This demands for example the consideration of more application fields (available through integration) in the control approaches.

1.2.4 Smart Home Demonstration Objects

Smart homes are still often regarded as a future vision that will be practically available only in some decades of time. In practice, however, the technological and scientific foundations have long been developed so that a realization of smart homes is already practically feasible today. Although the commercial break-through of the smart home is still awaited, several demonstration smart homes have already been built. They illustrate different approaches to the concept of a smart home and also provide valuable information on the challenges as well as lesson-learned information for the design of upcoming smart home concepts and systems.

The following sections provide an overview of the best known real-world implementations of smart homes, according to the definition given in this dissertation.

PlaceLab

The *PlaceLab* [30] is a laboratory built by the House_n department of the Massachusetts Institute of Technology (MIT). The Department of Architecture research group there mainly investigates how people interact with new technologies and home environments or how voluntary subjects react to architectural designs. The PlaceLab is constructed in a residential condominium and equipped with a variety of sensing components that are used to monitor different environmental parameters as well as human behavior. Examples for such sensors include motion sensors to track the position of the user as well as sensors on appliances to obtain the context in which context they are used. The according monitoring user interface is shown in Figure 1.9

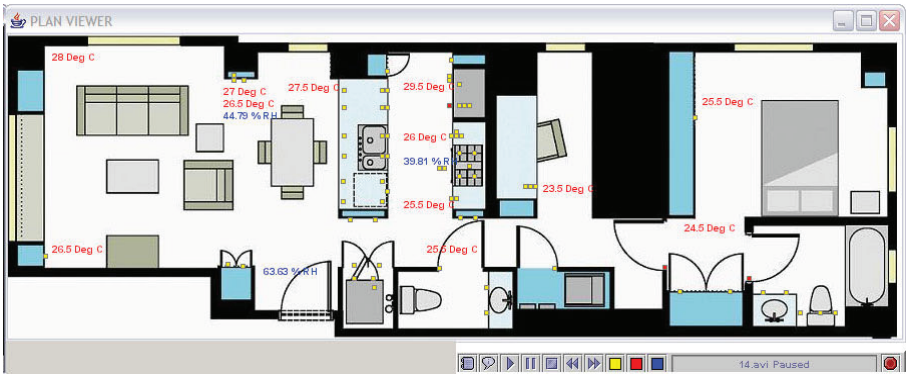

Figure 1.9: Sensor values measured in the PlaceLab [30]

The key research purpose of the PlaceLab is to develop key components for future buildings involving the manufacturers, builders, developers, designers and also the customer. Current projects that are pursued involve tracking and recognizing activities in daily living with ubiquitous sensors to detect medical emergencies, context-aware computing devices, portable low-cost wireless sensors for detecting motion and position and also medical values of the wearer (like heart rate or ultra-violet light exposure) or the development of ubiquitous computer interfaces. According to the homepage of the PlaceLab [31], MIT researchers use this incarnation of a smart home to find answers to the following questions:

- What influences the behavior of people in their homes?
- How can technology be effective in the home context for long time periods?
- Can technology and architectural design motivate life-extending behavior changes?
- To what degree can measurements of activity in the home be quantified in a way useful for creating new computer applications for the home?
- How can technology be used to simplify the control of homes of the present and future, save resources, and improve health?
- What influences how people adjust to new environments?
- How do people learn in the context of the home?
- What new innovations for the home would most fundamentally alter the way we live our everyday lives?

Aware Home

In 1998, the Aware Home Research Initiative (AHRI) was formed at the Georgia Institute of Technology. They built a model smart home in a common three-story house called "Georgia Tech Broadband Institute Residential Laboratory" that serves as a living laboratory for interdisciplinary sciences [32]. The house, which is known as the "Aware Home", is in operation since 1998, and research on and with it is still ongoing. Figure 1.10 shows an outside view and the floor plan of the Aware Home. Two floors of the house have the exact same floor plan so that they are used as two single-story apartments.

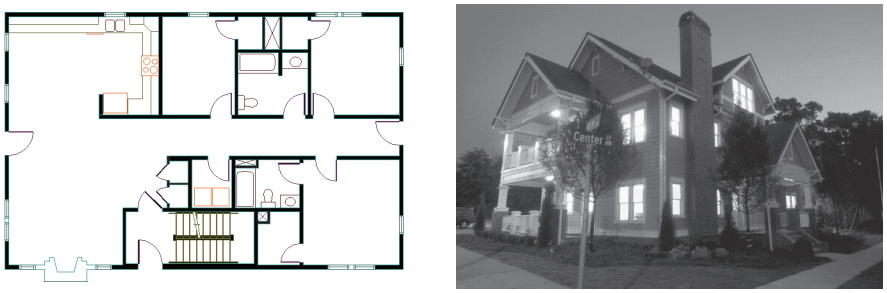

Figure 1.10: Aware home floor plan and outside view [32]

Research conducted in the Aware Home includes several domains. Applications mainly target the support of residents, differentiated in individuals and families, in different ways. Hence, the researchers focus in the directions of elderly care ("successful aging"), support of busy families and children with special needs, and the facilitation of daily routine tasks. As a second pillar, the Aware Home is used to develop and test new sensing and perception technology. Goal of this research is to increase the system's awareness of people's actions and activities. Additionally, AHRI works on the advancement of "Digital Entertainment and Media". Here, the research subjects range from usability and user interface design to security and privacy considerations.

T-Com Haus

The *T-Com Haus* [33] was a temporary smart home project operated by the German telecommunications company Deutsche Telekom. As shown in Figure 1.11, the house was on public display in Berlin in 2005 and intended to showcase the possibilities introduced by automation technology and a system of networked devices. The T-Com Haus was also used as a laboratory where interfaces and interaction models could be studied. Furthermore, interested people could live in this smart home for some time and experience its features and novel services.

Figure 1.11: T-Com Haus in Berlin (c) Jochen Jansen, Wikipedia

The building automation expertise in the house was contributed by Siemens, with the particular notion that all the deployed devices and technologies were already available on the market to end users. However, they were combined into an overall concept and used to demonstrate novel use cases in households. Integration of the heterogeneous devices was claimed to be achieved using solely the UPnP standard. The targeted domains were information representation (e.g., e-paper on demand delivery or online storage of virtual documents), multimedia (e.g., mood management to control lighting, audio and video devices according to the user's mood), entertainment (e.g., delivery of video/music on demand, virtual training courses) and security. Building on a so-called "home automation platform (HAP)", access to all devices and services over a central controller, in this case a PDA, was proposed. As examples of the monitoring and control possibilities, PDA based control of shutters, the heating or lighting system was mentioned. Additionally, status checks of various appliances and entertainment devices as well as the execution of configuration tasks were addressed.

inHaus

The German *inHaus* or *inHouse* [34] is one of the European flagship projects concerning smart home research. It was built and is maintained by the Fraunhofer-inHaus-Zentrum which is a subsidiary of Fraunhofer-Gesellschaft. The smart home was co-financed by several industrial partners with the automation companies Berker and Hager among them, and hosts research and demonstration activities of seven Fraunhofer institutes (covering the research domains of e.g., building physics, logistics, microelectronics, software engineering as well as energy systems and technology). Its aim is the joint research and development of commercially exploitable technologies in the areas of [34]: building and planning with support from IT, energy transparency and energy efficiency, logistics and operating processes, interaction between people and technology, multifunctional component building systems, sustainable construction, performance-oriented rooms, security and safety as well as electronic systems for assisted living.

The inHaus-Zentrum is distinguished into two separated projects: the inHaus1 building that covers residential buildings (SmartHome) and the more recent inHaus2 facility for commercial properties (SmartBuilding). The inHaus1 (cf. Figure 1.12) accommodates a living room, bath, kitchen and workshop laboratory and is equipped with a broad range of devices. Main research topics are the integration of heterogeneous systems, and the exploitation possibilities of a smart home environment

for Ambient Assisted Living and energy efficiency applications. Some examples are tele-medical appliances in the sleeping and bathing area and the use of intelligent sensors for optimal airing and air quality, as well as individual water management in the bathroom. Additionally, an "intelligent garden" with a networked watering system and an automated lawn mower and a multimedia car are available for research. Despite the initial intentions to reserve the building for research purposes it today also serves as a showcase for (commercial) smart homes. Still, its primary purpose is to support the development and testing of new products in the sector of home automation and smart living. As such, acceptance workshops and test living phases with volunteers are conducted in inHaus1.

Figure 1.12: inHaus 1 – Smart Home [34]

Thanks to the positive reception of inHaus1, the inHaus2 project was set up in 2008. According to the Fraunhofer-Gesellschaft, "the primary goal of inHaus2, shown in Figure 1.13, is the future-oriented development and commercialization of novel, intelligent room and building systems (Smart Building) to increase the overall market appeal of commercial buildings. These solutions support and enhance the entire lifecycle of a commercial building by optimizing its operating processes, from planning and construction to operation and facility management". The experience and technologies that proved well in the first house were adopted directly and complemented with further systems and recent technologies, such as an absorption-based air conditioning system. The second building sets its application focus on hotel business, healthcare and office labs instead of the residential area. inHaus2 is also characterized by a futuristic interior and exterior compared to the very traditional

architectural concept of inHaus1. Despite its stronger orientation towards a commercial market and exploitation, also research is conducted in this smart building. Several research results have been published, among them work on AAL solutions [35].

Figure 1.13: inHaus 2 – Smart Building [34]

Related both to the inHaus projects and this dissertation, the AMIGO project [36] is to be mentioned. This project is a large scale EC funded research endeavor to develop "Ambient Intelligence for the networked home environment (AMIGO)". During its course, a middleware for a networked home that enables interoperability among applications and services in the home was developed. The middleware has been tested and vastly exploited in the inHaus demonstration objects.

iHome-Lab

The iHomeLab [37] is a Swiss research laboratory for building intelligence and advertized as think tank. The smart home (cf. Figure 1.14) was created by the Lucerne University of Applied Sciences and Arts and is used to implement and evaluate results of adjacent research projects. Furthermore, it was built with the goal to increase user acceptance of smart homes and intelligent living by ways of public events and workshops.

An exhaustive list of applications that are tested and implemented in the iHome-Lab is given as follows: Advanced metering infrastructures, smart metering and load management, human machine interfacing, ambient awareness and AAL, home net-

Figure 1.14: iHomeLab exterior view [37]

working technologies, the Internet of Things and plug&play, information retrieval and artificial intelligence. Research in this area is conducted under the theme of "Buildings as System", which denotes an interdisciplinary, comprehensive approach in smart home research. As shown in Figure 1.15, the core application areas are energy efficiency and ambient assisted living. These two topics along with wireless systems and technologies (localization) as well as plug&play concepts also form the main pillars of current research projects.

Particularly interesting is a project on load signatures of appliances where the load curve on the power grid is used to identify active devices in the home. Another project takes a decentralized approach to measure energy consumption and control appliances with the help of a so-called *energy measurement dimmer* which also implements rudimentary load shifting algorithms.

SmartHOME

The Institute of Sensor Technology & Measurement Systems of the Bundeswehr University Munich operates a building called SmartHOME [38], in which intelligent sensor technology is used to improve energy efficiency, living quality and security of its residents. In contrast to other projects, the home is not intended for daily living but used solely as laboratory. The house is constructed following current low

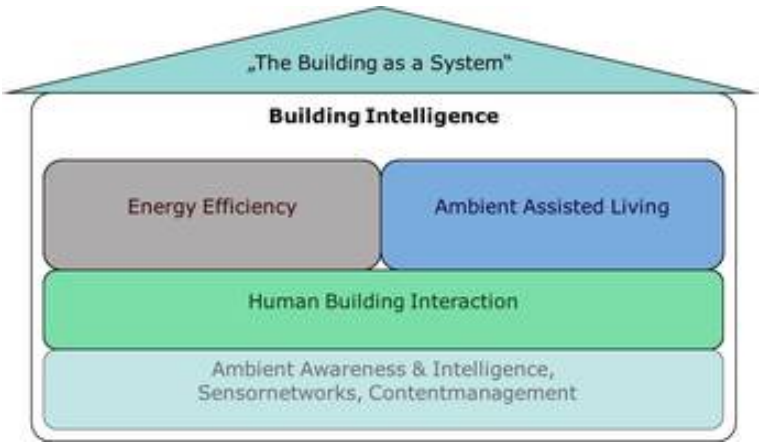

Figure 1.15: Buildings as System approach [37]

energy building standards with e.g., high insulation. Research therefore goes into the direction of enabling technologies for zero- or even plus-energy buildings and towards the efficient integration of renewable energy sources. For this purpose, a networked measurement infrastructure was developed in a first project step. In a follow-up project beginning in 2010, now control strategies to minimize heating energy and maximize indoor air quality and thermal comfort are developed. Challenges in this research include the establishment of a good air quality in buildings with almost no natural ventilation available, such as it is the case in passive houses. Through the application of model predictive control (MPC), fuzzy control or neural networks combined with simulation (cf. Figure 1.16), the researchers aim at solutions with high user comfort and acceptance [39].

Gator Tech Smart House

The Gator Tech Smart House [40] is an extension of the Mobile and Pervasive Computing Laboratory of the University of Florida. In this lab a preliminary mock-up version of the smart home was created and used for experiments, before a real home was built within a retirement community in Gainesville, FL. Accordingly, research focuses on technologies and applications for elderly and disabled people, such as assisted living systems (cf. Figure 1.17). The smart home is partly inhabited by human test persons that actively take part in the research experiments.

Most applications developed in the Gator Tech Smart House are related to the project ADI (Aging, Disability and Independence). Among them, in particular the

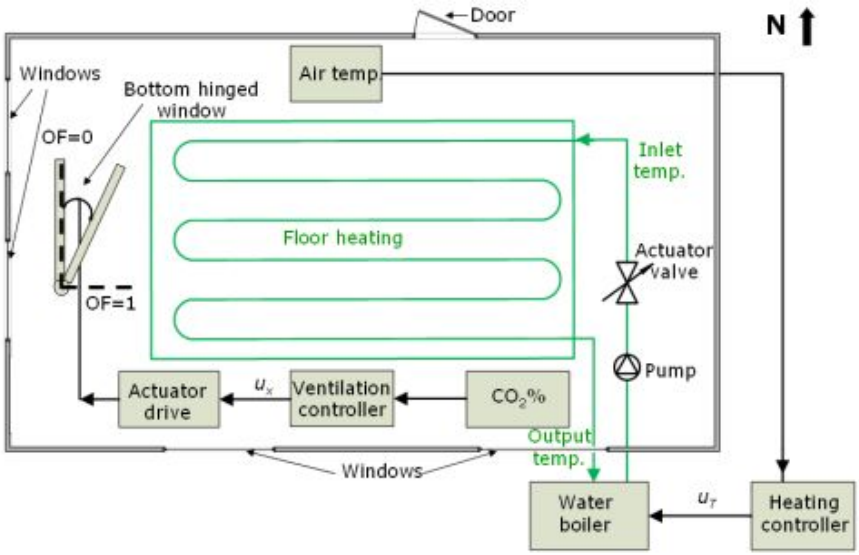

Figure 1.16: Heating and ventilation control system of a test room in the SmartHOME [38]

Smart Floor and the Smart Plug [41] propose interesting concepts. The Smart Floor targets the localization of people as well as some kind of activity awareness. Its main benefit is that it works unobtrusively, requires no user interaction and is invisible to the user because the sensors are embedded in floor tiles. The information provided by the pressure sensors is used in applications, such as the observation of physical activities (e.g., a step counter) or to detect occupant falls. The Smart Plug is shown in Figure 1.18. Through the use of RFID tags on the device plugs and RFID readers in the outlet, each electrical device in the house can by identified and localized when plugged into the electric circuit and be thus integrated into a smart home system. Combining this knowledge with remotely on/off switchable outlets, applications in the smart home can also control the status of not (yet) networked devices, such as coffee machines, lamps, fans or other electric devices.

Conclusion on Related Smart Home Projects

The discussed projects underline the increasing interest in smart home technologies over the past two decades and illustrate the potential benefits of such automated dwellings. They also demonstrate the practical feasibility of research results and allow to estimate the possible benefits of specific measures. Especially the latter point is a very important argument for consumer decisions and hence the availability

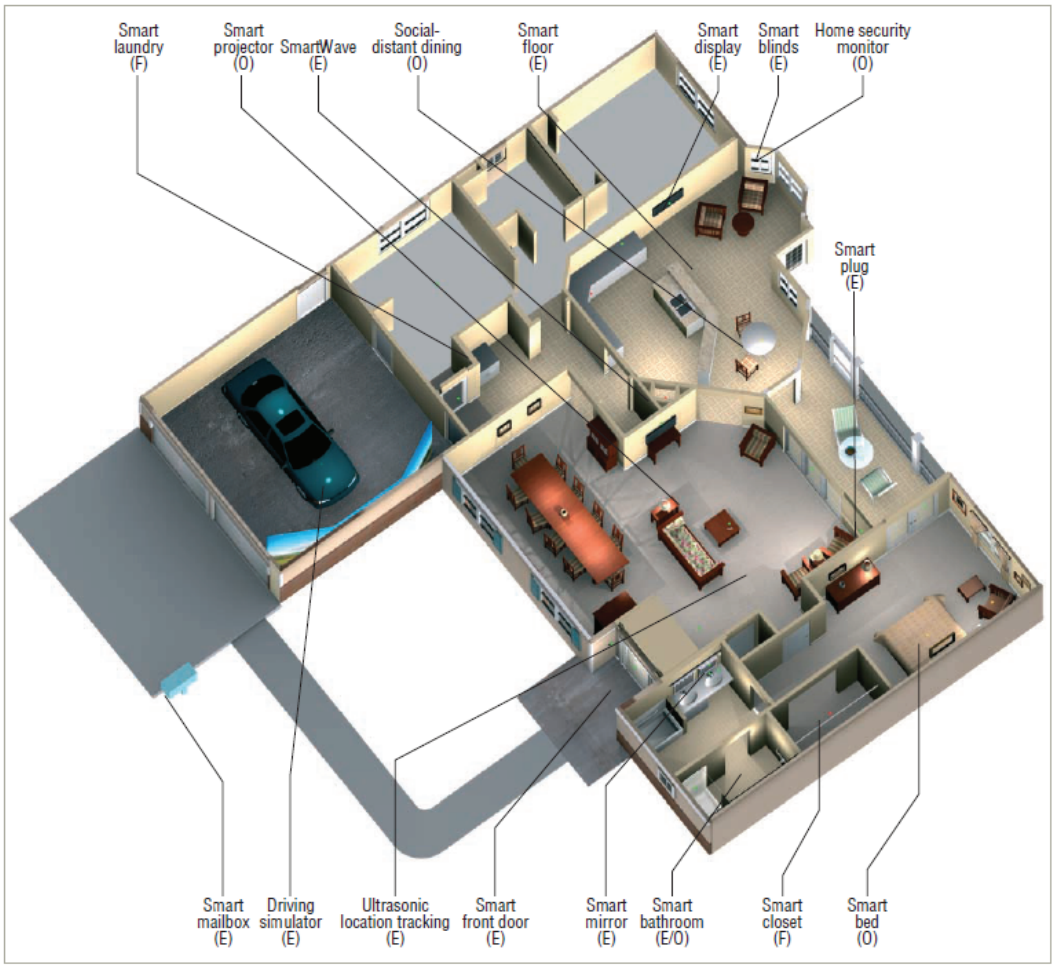

Figure 1.17: Floor plan and application overview of the Gator Tech Smart House [40]

of industry level smart home solutions. Not last, most smart homes can be visited and the interested public can experience smart home technologies directly, allowing both research to learn from these encounters and vice versa. Finally, these objects also have a high value concerning media presence and positive press articles thereby increasing the reputation of automation technology.

However, a shortcoming is that most projects concentrate on particular issues of smart homes only, e.g., they demonstrate approaches that increase energy efficiency or improve user interaction, but seldom they provide comprehensive concepts. Still, several ideas can be gathered from the projects and their approaches can work as guidelines for future developments, such as the comprehensive smart home system

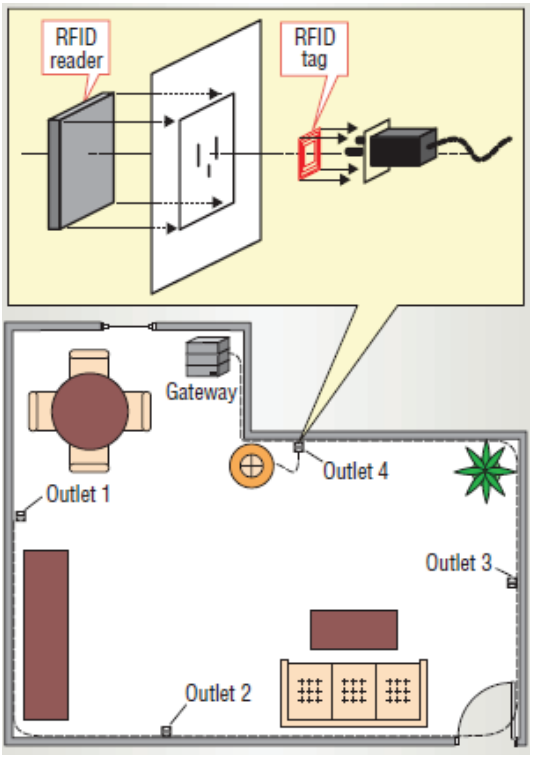

Figure 1.18: Concept of the Smart Plug [41]

that is developed in this dissertation. In Chapter 3, several ideas that have (to some extent) already been addressed by these projects will be reconsidered and partly re-used to form the theoretical concept of a new smart home system concept.

Chapter 2

Requirements of Smart Homes

This chapter aims at defining the future direction in which smart homes should evolve. It starts with an easily understandable description of the practical use of smart homes in their daily operation. Method of choice are user stories which depict typical scenarios which occur frequently in a smart home setting. They capture typical system actions and interactions in the smart home without going into technical realization details but give a particular focus on the contributions and functions of the (future) smart home system that ease the daily life of the user. The user stories set the scene for a detailed analysis how the challenges of smart homes (cf. Section 1.2.3) can and should be addressed in a smart home system. The approaches how the challenges are tackled result in a set of building blocks and a set of requirements that the future smart home system must implement. Again, concrete technical means how the requirements can be reached are not yet specified. Based on the informal description of smart home functionalities and the more formally captured requirements of a smart home system, a smart home system concept is presented in the final section of this chapter. In this comprehensive concept, the main system parts and functionalities that enable the operation of a smart home under compliance to the requirements specified before are identified and described. The resulting abstract system architecture is the main outcome of this chapter.

2.1 Smart Home Scenarios

A good starting point for a definition of a new system, such as the *smart home* are user stories. These user stories provide an overview of the system operation in a daily context and under strong involvement of the user perspective. Originally, user

stories were developed for the use in the field of *extreme programming* as summarized for example by Beck [42]. Today, ancestors of this programming style, such as agile software development still use this concept as a main building block in their design phase as for example Cohn points out in his book "User Stories Applied: For Agile Software Development" [43].

User stories are an *informal* description of the services that are offered by the smart home and of the interactions of the stakeholders with the smart home. They offer (not necessarily complete) scenarios that illustrate the "smart home in action" or at least particular relevant aspects of it and put the added value for the user into the foreground. Moreover, they focus on showcasing the possibilities of a smart home but stay decoupled from any technical implementations. Thus, they can be used as a tool to establish a common vision and understanding of the system (in this case the smart home) and what functionalities and services it offers.

In this dissertation, two different user stories are developed and described. They are constructed around typical daily activities in the life of fictive inhabitants of a smart home. The stories are centered on the functionalities in a smart home and described in a non-technical way, leading to the desired "narratives" more than technical specification documents and language.

2.1.1 User Story #1: Thermal and Visual Comfort Throughout the Day with Prediction

Alice is woken up by the alarm clock at 6:30AM. Previously, the smart home has gradually adapted the blind lamellae so that the natural light can slowly traverse the bedroom. Since the alarm clock is integrated into the smart home system, the heating system is started just in time before the alarm rings so that the preferred wake-up temperature of Alice is reached in the bedroom and adjacent bath. Therefore, the system has previously calculated the optimal starting time and set points of the heating system under consideration of current indoor and outdoor climates, weather prediction and the building structure parameters it has available. The pleasant temperature of Alice was learned by the system over a recent time span and is adapted whenever the system detects thermal discomfort in Alice. Once Alice leaves the bedroom for the kitchen to get a cup of freshly brewed coffee, the smart home turns off idle electric devices, such as the radio in the bathroom that was switched on by her while having a shower. It also checks the air quality in the bedroom and the humidity in the bathroom to decide if airing is required. Although the air quality

sensor in the bedroom indicates a CO_2 level above the threshold but below a critical value, the smart home decides to postpone airing.

Meanwhile the heating system is re-programmed to keep only a lower set-back temperature from now on. Still, Alice feels comfortable in the kitchen while having her breakfast since the temperature does not decrease immediately in her well-insulated home – a fact that has been observed and learned over time by the smart home. Alice is now ready for work and leaves the home through the garage in her car. Once the car has left the premises, the smart home checks for closed doors and whether the oven is switched off. After the successful assessment of the safety status of the home it sets the safety indication flag of the home on the private Web portal to a green check mark.

Now, the outside air temperature has reached a similar level than the targeted indoor temperature. Hence, the smart home decides to execute its daily airing strategy and chooses natural ventilation among its pool of airing options to fulfill this goal. The smart home opens the windows on the upper floors and tilts selected ground floor windows to achieve a natural draft. Simultaneously, the intrusion alarm sensors of the areas with open windows are activated. In the morning the smart home has also received the hourly weather forecast for today that predicts moderately warm temperatures and sunshine for the whole day. As it has not rained for the last two days and no precipitation is expected for today the smart home decides to water the grass in the garden right now so that Alice can enjoy a dry grass in the evening.

In the late afternoon, Alice uses her mobile phone to open the smart home application and send a reminder to her smart home that she will have dinner in town and will arrive home later this evening. Immediately after the reception of this message the smart home assesses the previously planned operations that are connected to the presence of Alice, such as those that ensure comfort for Alice. Because the heating system was scheduled to start with respect to the typical arrival time of Alice, it is automatically adapted to match her expected arrival time.

In the evening, Alice returns home after a pleasant dinner in town. Upon her arrival, the smart home activates the welcome home scenario which provides a pleasant lighting and soft music in the background. Furthermore the TV is switched on, muted and the channel is set to the evening news. Next to it, the smart home uses some space of the TV to inform Alice about a missed parcel delivery and provides her with a link to a website with further information. By means of a small chart Alice is also informed about the energy consumption of the day. Once Alice sits down on the living room couch in front of the TV, the TV lighting scenario is started, the living

room lights are dimmed accordingly and other devices are deactivated in order to maximize comfort for Alice.

2.1.2 User Story #2: Thermal Comfort Throughout the Day with Prediction

Bob and Cate sit together for breakfast in the living kitchen of their modern apartment. It is a cloudy day in late April and the weather forecast predicts of 40% chance of rain around midday. Hence, Bob and Cate decide to take the car to get to their work place. Because Cate favors a sustainable lifestyle, they own an electric car that is parked in the garage right below their apartment. Over night the smart home has taken care that the electric family car was fully charged using the current night tariff promotion of a local energy provider.

Bob and Cate's apartment features a large terrace on which they have recently installed some photovoltaic panels. Although the weather forecast predicted an overcast sky, the sun shines on the panels and electric energy is produced. The smart home notices this condition and starts deliberating on the best strategy to use this excess energy. Since the feed-in tariffs of energy are low the system decides to use the energy to operate the geothermal pump which belongs to the building. The heat derived from it is then used to pro-actively heat the apartment to higher temperatures than the current set-back temperatures so that in the evening less heating energy will be required to guarantee comfortable conditions for Bob and Cate.

In the morning, Cate loaded the dishwasher and programmed it for a starting time at noon. However, at 11AM the smart home is informed from the energy provider that a voluntary opportunity for a demand side management operation is available because the provider faces an unexpected electricity consumption peak around noon. Upon this notification the smart home internally checks its load shifting possibilities and starts interactions with the energy provider. It is mutually agreed that the operation of the dishwasher will be delayed for one hour in exchange with financial incentives on the energy prices for today. Thus, the smart home re-programs the dishwasher to a new later starting time since it knows that Bob has specified in his preferences that load shifting shall be accepted whenever it does not violate any of his other preferences, such as clean dishes when he arrives home in the evening.

In the afternoon, the cleaning person arrives. The system detects the presence of a person and identifies this person as authorized guest. From this guest's preferences, the smart home knows that warm water is required for cleaning and thus activates warm water production. After some time the guest starts to clean the windows and

therefore opens them. Then, he/she moves on to the bathroom and forgets to close this window. After several minutes, the smart home detects that nobody has been in the living room for a longer time but that the window is still open. Since the outside temperature is below the desired indoor temperature, the smart home informs the cleaning person that the window should be closed to conserve heating energy. The person complies with this request since he/she forget to do so before and the heating strategy is resumed.

In the late evening, Bob and Cate go to bed. Once they settle down in their bedroom, the smart home starts the sleeping mode where all doors and windows are closed and locked if not done yet and the alarm system is activated. Additionally, noisy operations in the home are prevented from this time on as far as no immediate need is reported from the sensors. Furthermore, the smart home disconnects all electric devices that do not need permanent power supply, such as the coffee machine or the TV from the electric grid to follow energy saving preferences and to increase safety. However, the smart home notices that Bob has programmed the home entertainment server to record the rerun of his favorite TV show. Thus, it automatically schedules the power supply for the home server to be activated safely around the airing time of the show.

2.2 Addressing the Challenges of Smart Homes

One main point in the concept of a novel smart home system that is ready for the 21st century is that it must address the current challenges that exist in smart homes. These challenges were listed and discussed in detail in the previous chapter of this dissertation. In the following requirements list, special characteristics and solution approaches that a novel smart home system should provide to alleviate the aforementioned challenges are described. Thereby, these new requirements are not a direct (i.e., 1-to-1) mapping from the challenges in Section 1.2.3. This is mainly because sometimes more than only one new requirement needs to be introduced to address a particular challenge. Likewise, one of the requirements will often contribute to (at least partly) tackle several challenges at the same time. For an easier mapping between challenges and requirements and also for better overview, the addressed challenges are references in brackets after the requirements.

- Integrated system with a central control and management point (*Integration, Homogeneous system, Energy efficiency and comfort, Economic benefit*)

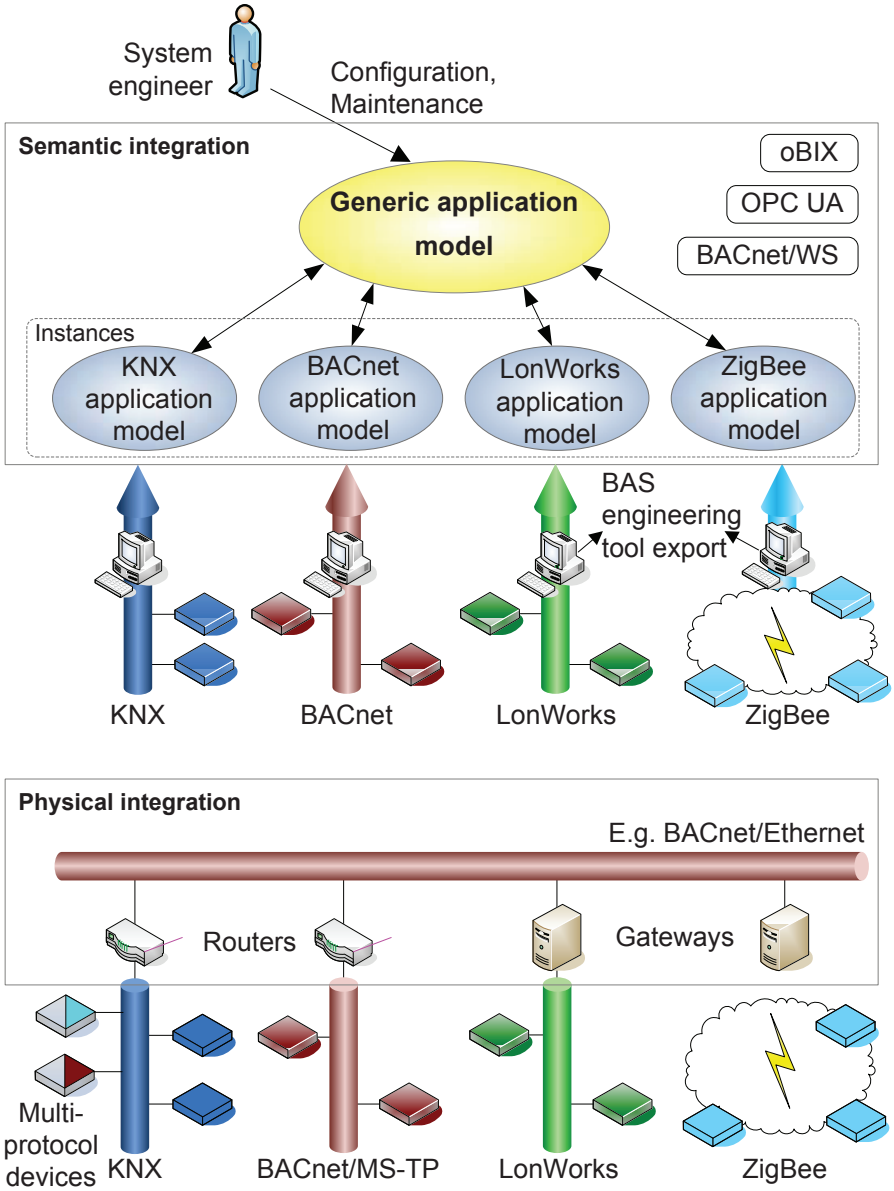

Figure 2.1: Integration approaches at physical (lower part) and semantic level (upper part)

In today's homes a multitude of different devices are used, many of them running on different protocols. In a smart home, all these devices must be connected to each other to allow for their integrated control and to enable advanced control strategies. This requirement is not unlike the vision of the

Internet of Things (IoT) [44, 45], where physical and virtual objects shall be seamlessly integrated. However, the IoT also encompasses additional domains, such as supply chains, logistics and electronic product codes which are not much related to smart homes. The performance of a smart home system in terms of control should be untouched by the kind of underlying networks, i.e., the control should be agnostic of underlying technologies[1].

From a technical viewpoint, integration in the smart home demands both the physical interconnection of the devices as well as the semantic interoperability of the exchanged data. Especially the latter requirement is not fulfilled easily, yet of major importance. For each existing protocol, devices that are proven to comply with the particular technology are able to communicate with each other – even if they are sold by different vendors. This vendor-independent compliance is called interoperability. Interoperability not only has to be guaranteed for communication but also for distributed applications, so that devices of different vendors can be deployed in one system. Therefore, all technologies define their own application model which states how data is represented (data format, encoding) and how the communication between applications has to be realized (methods to manipulate data). In practice, application models are specific to particular technologies, which prevents direct communication across protocol borders. However, in the smart home, interoperability is no longer confined to protocols but must be guaranteed in the whole integrated system.

Both of these demands (i.e., physical and semantic interoperability) can be realized by the use of gateways or routers that interconnect the different, protocol-specific network segments to a common backbone as shown in Figure 2.1. However, a common, generic application model must be devised first before it can be implemented in the gateways. In the lower part of Figure 2.1 the physical integration approach is illustrated. Depending on the media dependent OSI layers (i.e. physical and at least part of the data link layer) specified by the different protocols, dedicated devices, such as routers or gateways must be employed for network segment interconnection. While routers are less complex than gateways, the latter devices provide more functionalities, such as an application layer mapping but are accordingly more complex concerning configuration and maintenance. In their practical use, however, both approaches realize a physical aggregation of the different protocols as

[1]The number and type of deployed devices or subsystems (e.g., the domain coverage) has, however, an undeniable influence on the control possibilities.

all exchanged information becomes available at a common backbone network segment. Due to the large data amount encountered there, this backbone is mostly Ethernet and IP based today. If a communication across protocol borders is targeted as it is frequently the case in smart homes, also multi-protocol devices can be used. These special devices implement two or more protocol stacks of different technologies in parallel and can thus communicate using all technologies they have a protocol stack for. While this automatically enables the use of one device and its datapoints in multiple protocols, such devices are characterized by considerable overhead in production and costs and thus not frequently used today [46]. The upper part of Figure 2.1 shows the necessary semantic integration approach of heterogeneous systems. As can be seen, each technology has an own specific application model, thus a homogeneous view on these technologies can only be provided if also these differences are abstracted. This can be accomplished by the specification of a generic application model and the according mapping of the technology specific models to the generic one. In general, this is not always a straightforward approach but requires intensive knowledge of technology specifics. The mapping that is later uploaded to the interconnection devices (i.e., gateways) is hence defined by integration engineers. Currently, a trend towards an integration with the help of Web service technologies, such as oBIX, OPC UA and BACnet/WS can be observed, where a framework for integration and information access is provided. One way to facilitate the semantic integration process is to rely on configuration data that is exported from the original technology-specific configuration tools, re-arranged and then used for gateway configuration. Once the integration process is accomplished, the smart home can offer additional functionalities such as described below.

A smart home system should provide a central point where configuration and management tasks can be performed. It must assure homogeneous system impression to applications, users and external services, such as demand side management, the smart grid or service providers in domains, such as performance contracting and security/safety. Therefore, it is necessary to abstract the underlying protocol and system details and provide a generic view on devices and services. The integration approach must thus tackle physical layer, data transfer and communication as well as application model differences. Moreover, it should provide a generic and easy-to-understand and easy-to-use interface to

the underlying technologies, thereby abstracting any specifics of protocols and automation systems.

With a central control and management point or interface, also energy efficiency and comfort benefits can be yielded. On one hand, the whole smart home system becomes (theoretically[2]) accessible for the outside world. In particular, this is an interesting perspective for home owners who can access their smart home worldwide over the Internet and perform monitoring and control tasks remotely.

On the other hand, the – in the best case standardized (e.g., Web services) – interfaces to the smart home system can also be used to offer new services to smart home owners. (Re-)configuration assistance can be offered by specialists located anywhere. Companies can offer performance contracting services where part of the control authority is handed over from the smart home inhabitants to external services that mostly promise economic benefits in exchange. One example of such an application is demand side management, where energy consuming devices are (partly) controlled by an external authority, such as the energy provider itself [47].

Not last, a central control and management point offers the possibility to abstract unnecessary complexity and provide tailored interfaces to different smart home stakeholders. Foremost, this is highly important for the smart home inhabitant who will profit from easy-to-understand user interfaces that offer a desired amount of complexity to the user and abstract details, such as the heterogeneous communication protocols that are used to connect the different devices together. The achieved global view of the system where all information is aggregated can be further exploited for data analysis. Energy consumption data can be collected and analyzed to present consumption graphs and statistics to the user. It is also possible to present timely feedback, such as energy savings tips on user actions.

- Learning capabilities (*Adaptability, Usability, Increased energy efficiency and comfort, Economic challenge*)

  Smart homes must have a perceivable added value for their users. One main aspect of this is the requirement that the system must be able to adapt to its environment and hence also to its users. In the context of a home, many

[2]Practically, precautions will be taken that ensure privacy and security of the smart home system and its functionalities, thus regulating the accessability to the smart home.

aspects can change over time. Rooms are remodeled or the room usage changes, different numbers and different persons inhabit the home, exterior influences change and even the building structure may be adapted.

In contrast, home (and often also building) automation systems are in many cases installed and configured only once. Later changes to the configuration of the automation systems are mostly foregone simply due to lack of knowledge or also because of the high costs encountered when specially trained configuration staff is needed for adaptations. This inevitably leads to sub-optimal control performance combined with an unnecessary financial loss and very often also a personal dissatisfaction with the system.

Thus, a modern smart home system needs to provide functionalities that alleviate this shortcoming. A possible solution is found in learning algorithms combined with context awareness. This way, for example, the user is relieved from actively telling the system his/her preferences but the system learns them from behaviors or through data mining of sensor values. Also in case of a changed usage type it should be sufficient to inform the system about the new room usage types. The system should then automatically infer the required changes in its control strategies (e.g., a different basic heating schedule for a children's room than for an office). Such an intelligent behavior has implications not only on the subjective comfort feeling, but should also result in reduced operational costs.

- Advanced control strategies (*Increased energy efficiency and comfort, Integration of user, Economic challenge*)

  In present day automated homes, the potentials of energy savings as well as comfort are not fully exploited yet. One main reason for this is that control strategies have not kept up with the progress and been aligned to the increased range of functionalities, devices and general information sources such as the World Wide Web that are available today. Additionally, with the integration of previously unconsidered domains, such as entertainment devices, household appliances or energy producing facilities, automatically more information becomes available in the system. Not least, the advance of computing technology in almost all domains facilitates the (re-)use of data that previously was created and exclusively used by specialists belonging to other building related domains, such as architects, building engineers, building physicists or facility managers. The integration of their collected data can be a valuable source for advanced

control strategies. In the scientific world the latter approach is researched in a field called Building Information Modeling (BIM) that has gained also considerable commercial interest in the past years. The goal of BIM is the definition of a comprehensive model that is consistently used from the design phase, over construction, until the operational phase of a building [48]. Main idea is the stepwise addition of data to the BIM by the different experts so that, unlike a couple of years ago, information is no longer lost or confined to specific disciplines. However, BIM is a rather collective term that summarizes a range of different computational building models that differ in their complexity and coverage of design steps [49].

The information contained in a building information model, the possibilities arising with the integration of additional devices and domains as well as the availability of the WWW as vast information source are the main enablers of novel control strategies in smart homes. However, another requirement next to this *information accumulation* is to represent the knowledge in a way so that it becomes usable for the smart home system. In particular the algorithms that are executed require access to the information, thus raising the question of *information representation*. Only if both requirements are fulfilled, the advanced control strategies can be successfully implemented and executed in the smart home system.

The identification and specification of such novel control strategies is a task still to be solved in research. First approaches, for example model predictive control using weather forecast data [50] or sensor fusion for activity recognition [51], showcase the possibilities of an integration of additional information sources and can work as role models for future developments. The conception of the algorithms will also be guided by theoretical analyses of optimization potentials as well as draw much input from the knowledge and best practices of human domain experts.

Out of these building blocks, new algorithms will be composed that ensure even more energy efficiency and comfort for the inhabitants of the smart home. The benefit is that control strategies can be designed that would be too complex or even impossible to be executed manually by human users but are comparably easily executed by computers such as the smart home system. The limitations today are, for example, introduced by strategies requiring rapid control decisions, the consideration of large amounts of data or simultaneous control of a

large number of devices. Finally, a requirement of the control strategies is to design them in a way so that they support but never patronize or overrule the users [52].

New control strategies should incorporate both static and dynamic data towards a better performance. Static data includes the information stored in the BIM, such as the orientation of buildings, the characteristics of the building hull (e.g., insulation, openings) or the floor plan as well as data from the automation systems, such as devices and device functionalities. Dynamic data, as the name implies, is likely to change over time but nonetheless prone for consideration in the control strategies. The latter data class includes weather predictions, the current status of devices, all sensor measurements and actuator states, occupancy, user activities and user preferences.

- Availability of additional information (*Increased energy efficiency and comfort, User integration*)

  This requirement is closely related to the demand for advanced control strategies. The underlying control philosophy is that the more information is available, the better the control strategies can perform. Quality in this case on one hand touches a quantitative aspect, i.e., measured in terms of energy savings or financial benefit and has a qualitative dimension, i.e., measured as user satisfaction, degree of adaptability to environment/user or similar aspects.

  The smart home system achieves better control by the execution of control algorithms. These algorithms require different input data depending on the type and complexity of the algorithm. The minimum set of information to be made available in a smart home system is thus determined by the input parameters of the control algorithms. However, in order to develop a future proof solution it is advisable to consider more than only the minimum data. This is for two reasons. First, more elaborate algorithms might become available and integrated in the smart home system in the future. If their required input information is ready to be accessed, the exchange of algorithms (and thus smart home control approaches) is facilitated. Second, available data might not only be used as input for control strategies but can also be the basis or enrich other parts of the smart home system. This concerns for example user interfaces that are presented to the user. In this case, it can be desired to visualize the floor plan including windows and doors (i.e., construction data) in the user interface. Additionally, some data that is not directly used in control

strategies can nevertheless be of interest to (external) service providers, such as DSM, maintenance personnel or performance contractors.

Concerning data availability it is not only important what kind of data is offered but also how this data is stored (*data storage*) and how it can be accessed (*data access*). In other words, the mere availability of information is not sufficient for an optimized smart home control. Again, control strategies are the main driver for the second part of the requirement as they shall be executable autonomously by the smart home system without user participation. Therefore, all data that is needed must be provided in a first stage for machines and hence be available in a machine-readable and machine-processable way. This special case is one of the few times when a "computer" requirement obtains a higher priority than a user's one. Still, also the human user shall be able to use the data (e.g., for any kind of analysis). However, to fulfill the latter use case, software is required (or at least advisable) that translates or modifies the stored data into a form that is easier readable/usable by humans. A simple example of such a piece of software that processes the data is a (graphical) user interface. Apart from data access, also the issue of data storage must be addressed. Here, the main goal is to find a technology that allows to store large amounts of data, provides reliable storage facilities, allows the alteration of stored data and is at best characterized by acceptable data access times.

Finally, the basic requirement of data availability can be extended towards further usage scenarios of the stored data. The idea is that once a data store is available in the smart home system it could not only be used as a passive component but also provide active features for data analysis and data interpretation (*data mining*). This would for example allow to draw conclusions from the stored data, among others by exploitation of relationships between data sets. Possible applications for this include the creation of trends, the automatic classification of data or the computation of predictions. In this way, it becomes possible to add another layer of intelligence in the smart home system.

- Automatic/autonomous control (*User integration, Economic challenge, Usability, Increased energy efficiency and comfort*)

  One of the main contributions of a smart home system is that it shall ease the control of a home for the user. Therefore, it must take over routine tasks from the user, support him/her in decision making concerning control aspects and take care of the execution of selected control strategies. Automation can also

help to reduce the cognitive efforts for the users. This is of particular importance as soon as the control of very complex physical processes in the home is concerned. Equation 2.1 taken from [53] illustrates the calculation of the *heating energy demand* as an example. Obviously, the formula is considerably too complex to be used by home owners every time they want to adjust their heating system. An automation system could, however, use this formula as part of its control strategy.

$$Q_{HED} = Q_I - \eta_h * (Q_g + Q_{recovery}) + Q_H + Q_{tw} + Q_{TW} + Q_{HE} \qquad (2.1)$$

Q_I ... transmission losses
η ... utilization factor
Q_g ... heating gain from persons, solar radiation, etc.
$Q_{recovery}$ ... recoverable shares of the system losses
Q_H ... losses from space heating
Q_{tw} ... water heating energy demand
Q_{TW} ... total losses regarding the warm water supply
Q_{HE} ... auxiliary power losses

The tasks of the smart home system regarding the control requirement must be executed under two aspects. First, the smart home system must be characterized by a certain degree of autonomy. It shall be able to make control decision on its own and in particular with requiring as little user input as possible (e.g., it does not ask the user for a preferred room temperature). While this explicitly does not exclude any voluntary user interaction, it ensures that the user is alleviated of bothering routine tasks. Autonomous behavior of the smart home system also adds flexibility in the system operation as no longer completely pre-defined control paths need to be taken, but the system is equipped with some kind of variability in its decisions. Since safety is of major concern in smart homes, the system autonomy has to be seen within borders so that in practice not all decisions will be available for execution at a given point in time.

While autonomy targets more the control decision making part of the smart home, the adjective "automatic" is demanded here to guarantee that the previously made decisions are also executed without mandatory user interaction. In this case, automatic particularly demands that actions are taken without any

prior user initialization and also are executed without any required (human) participation or supervision. The second requirement of automatic control is normally ensured by the use of home (or building) automation systems that – as their name implies – have automation as a core design goal.

- Wireless technologies (*Retrofitting*)

  One requirement of a future system is that it supports wireless technologies. The main reason for this requirement is that wireless technologies are an emerging sector in home automation. In particular, the greatly reduced installation effort makes them an excellent choice to tackle retrofitting problems and thus increase the percentage of homes that are equipped with automation technology. Likewise, not only new installations from scratch, but also the extension of an existing, already installed automation system towards, for example, new application fields is facilitated.

  In context of the smart home system, wireless technologies can become a key factor for a more widespread use. Regarding the control strategies, cases will arise in a smart home system where with the addition of a few devices (e.g., sensors) many new control possibilities will arise. This small-scale upgrades are supported by wireless technologies that are quickly installed and can often be configured by the users themselves, which both contributes to less costs and effort compared to wired systems.

  From a technical viewpoint, wireless systems often have different operation modes and features (e.g., transmit-only devices, mandatory security, unreliable communication links) compared to their wired relatives. The differences mostly result from the support of low-power, battery-driven or energy harvesting wireless devices and from the inherent characteristics of a wireless communication channel. In the smart home system, the specifics and challenges of wireless automation technologies must be considered and addressed during system design to ensure the full integration of all technologies and systems.

- Universality (*Adaptability and modularity*)

  A smart home system must operate on different automation systems that are furthermore of different sizes and feature different equipment. One reason for this is that several different protocols exist in the smart home that are tailored to specific building functions and that offer tailored services for these niches (e.g., as DALI does for lighting control, where for example adjustable dim-

ming speeds can be configured). While services of these underlying protocols may work better (i.e., more efficiently, faster, more reliably,...), the concept of a smart home system must not rely on the availability of these advanced functionalities. Similar to this is the demand of the availability of certain sensors, actuators or controllers in the underlying system, which is also considered counter-productive especially regarding the broad applicability of a novel smart home system. One example for this latter limitation is the demand of automatic window openers. While these sophisticated devices may offer new control possibilities, they are not in widespread use yet. Therefore, classifying them as mandatory components would considerably limit the application potential of a future smart home system[3].

Nevertheless, a smart home system obviously puts some requirements on available services and equipment. In practice, there is a limited set of standard functionalities and respective devices that must be provided before one can build a smart home system on top. The requirement of universality is put on top of this minimum function set and shall ensure that a smart home system is not designed specifically around a particular automation system and/or set of devices. The system concept shall be universal and in some respects generic enough so that it can operate on many different underlying system(s) and equipment. In particular, as few functions and devices as possible should be classified as mandatory for a successful system operation. At the same time, the smart home system should be capable of exploiting as many non-mandatory (advanced or "luxury") features as possible and integrate them during its operation.

Taken together, both above requirements characterize universality. If these requirements are translated to a more system-design centric view, a system concept that adapts to the underlying hardware and software (i.e., automation devices and their protocols) is required. This way, a partly generic system concept can later be instantiated in accordance with the current situation.

Universality also plays a role regarding retrofitting. In this case, it shall ensure that also smaller installations of automation systems (which is often the case in retrofitting) can benefit from smart home control. If not the full spectrum of automation devices is available, intelligent control of building services is nevertheless possible, although one may not be able to yield maximum efficiency in

[3]However, it is likely that their availability would result in even better control performance.

this case. Additionally, universality as design requirement contributes to some system resilience against failures. Even if some devices or even a whole subsystem become unavailable, the smart home system is designed to also operate under these conditions and hence should be able to handle these failures and stay operational. This desired behavior is known as *graceful degradation.*

- Extensibility *(Economic challenge, Increased energy efficiency and comfort)*

  Similar to universality, the requirement of extensibility ensures that the smart home system becomes a future-proof solution. In case of later extension, the system shall not only remain operable with respect to the original functions but be capable of integrating the newly added devices and functionalities into its control. Extensibility also concerns changes in the user structure, i.e., (permanent and even non-permanent) users being added to or leaving the home, which shall be considered by the control approaches.

  When a system can be extended any time, this has also economic consequences. Vendor lock-in situations can be avoided and also little initial investments are sufficient for basic control optimizations. Likewise, targeted later extensions also pave the way for better control performance related with higher energy efficiency and comfort.

  Technically, later extensions shall be made as easy as possible. Ideally, plug'n'play concepts could support this tasks. Also central and in the best case technology-agnostic installation and configuration schemes (e.g., centralized configuration with automatic download to respective underlying base technologies and systems) facilitate later extensions. However, the latter requirements can only be met to full extent if both the underlying automation systems and the smart home system are designed and built accordingly.

After the in-depth analysis how the current challenges can be addressed and at least partly solved or alleviated, it becomes obvious that only a comprehensive system concept can provide the adequate basis for a smart home system. This approach must aim at combining the different requirements elaborated above into a homogeneous system. The motivation for a comprehensive system concept comes from the fact that if only a single requirement is fulfilled the full benefit cannot be yielded. A smart home system needs to combine many different parts (e.g., novel control strategies that operate on additional knowledge and are executed autonomously) so that the overall solution is more than the mere sum of its parts.

Furthermore, it is also not sufficient to fulfill each requirement in an isolated way as interdependencies between the requirements exist. Some requirements can also not be combined easily or at least not both requirements may be guaranteed to full extent. For example, user consideration is sometimes in contrast with an autonomous system operation. The problem can be targeted by considering both requirements at the same time thereby establishing trade-offs between them. However, this demands a comprehensive system design approach. In contrast to this, the isolated approach towards system design is likely to first produce several individual parts that are later brought together in a common framework. Often, these parts are not specifically designed with regard to an integrated solution but are intended to serve as stand-alone systems. Hence, the single parts may be powerful solutions if regarded on their own, their combination with other parts not always guarantees that the benefits still persist in the integrated system. Rather, it may result in a fragmented system with many different interfaces which again limit the performance of the system. As such, this approach of isolated design and later combination is similar to the current state-of-the-art in smart home systems where existing parts are taken and somehow brought together. One of the best examples for this shortcoming is the currently advocated combination of heterogeneous, often application-specific systems where the real benefits of the systems (e.g., support of special communication types) are often lost during their integration. Such an approach hence requires much effort but mostly offers only marginal benefits.

Therefore, a comprehensive system approach is considered as a mandatory criterion for the design of a smart home system. Although the decision of a comprehensive approach does not correspond to a typical requirement as listed before, it also contributes to tackle several challenges of smart homes. Among them, integration is one of the most obvious ones. An integrated view from the very beginning almost immediately goes along with sharing of information throughout the whole system. It thus enables applications such as novel control algorithms that consider cross-domain information or sensor sharing. A comprehensive approach also promotes the implementation of a homogeneous system and can contribute to a better user integration. Finally, a comprehensive approach is less likely to leave out important system parts. This is not last an economic advantage as any solution is in this case more likely to be future proof. The methodologically sound design of such a homogeneous smart home system is the main contribution of this dissertation. The comprehensive system for smart homes is termed "*ThinkHome*" to underline the importance of intelligent decision making and adaptability in the future smart home.

2.3 ThinkHome Concept

In a first step, a generic, high-level concept of a smart home system is developed. It shall illustrate the overall smart home system architecture and specify key components of a smart home system that are required to fulfill the system requirements. In the description of the architectural diagram, furthermore also the (high-level) relations and interactions between the components are captured. The presented architecture is still independent from any technologies or specific implementation approaches, but provides a conceptual view of the building blocks. It reflects the comprehensive system design approach that aims at ensuring all smart home requirements. Additionally, the architecture provides a classification of system functionalities and assigns these dedicated functionalities to two distinct system parts.

In a later step, the system concept and its different parts need to be instantiated by specifying technologies and developing concrete implementations. While the architectural overview shows several distinct modules of the future system, a realization may merge modules of the single layers or re-locate functionalities from one module into another. During implementation, the architecture hence rather provides a guiding point than it is a complete blueprint for system implementation. Nevertheless, its purpose is to preserve the comprehensive concept and its core functionalities also during system development.

The comprehensive concept of a smart home system is called "ThinkHome". The name ThinkHome shall underline the importance of the availability of intelligence in a smart home – a home that can "think" or deliberate about the users, building functionalities and further tasks of a smart home. The concept is derived under consideration of all the requirements stated in Section 2.2. Its design is in particular guided by two main premises: the smart home system shall ensure energy efficiency and simultaneously guarantee a desired user comfort in a home.

From an architectural viewpoint, the system is logically split into two main parts. Its schematic is shown in Figure 2.2. There it can be seen that, apart from the main system parts, also the *Global Goalsetting* is depicted. This shall underline the importance of a consideration of the global goals and furthermore illustrates that the smart home system shall always operate with respect to these goals. Consequently, the system was designed with these targets dominantly in mind, where both goals are represented explicitly and implicitly in the resulting system architecture.

Below the global objectives part, the two main system parts *Smart Home Control System* and *Knowledge Base* are shown. As the smart home control part, a software

system is designed that shall ensure the execution of the control algorithms in the smart home. The software part is furthermore responsible to handle user inputs, interface with the home automation systems, and to provide a user interface as well as further interfaces towards all kind of external systems. It shall moreover continuously ensure an operation of the smart home that complies with the global goals and user preferences.

Below the smart home control layer, an extensive knowledge base is located. The task of the knowledge base is to hold all data relevant for the system operation and to support the smart home control system in the execution of appropriate control strategies. Hence, it shall be capable of intelligently maintaining all relevant concepts that are considered to be influence factors in a smart home. In this respect, it is considered fundamental in grounding ThinkHome. On these data stored in the knowledge base, also data analyses shall be performed. These shall be based on approaches and algorithms known from knowledge engineering. Finally, the knowledge base shall provide storage facilities for historic data, which can be exploited for example for trending applications.

2.3.1 Smart Home Control System

The smart home control system is basically a piece of dedicated software that has the main task to ensure the daily system operation. It furthermore brings together the different systems and acts as a glue between them to realize a homogeneous system.

The software is (logically) partitioned into multiple components that each realize a set of dedicated functions. Taken together, all parts form the overall system. Depending on the technology chosen for an implementation of the architecture these parts may or may not be realized as independent software components. At the present stage, the parts hence represent a grouping of functionalities but do not yet specify and especially not limit the later implementation. Nevertheless, the ideas and functionalities provided by the different identified components may also be relevant in other context apart from smart home control. Hence, the components could also be taken and used to build or enrich other applications than exclusively smart homes.

Core part of the smart home control system are the *control algorithms*. The algorithms implement the core smart home functionalities, such as temperature control, lighting control or operation of appliances. Due to the heterogeneity of the underlying processes and the domain-specific requirements there will be multiple control algorithms in the system. Also, different automation equipment might require different control approaches and even different strategies (e.g., differing by their complexity

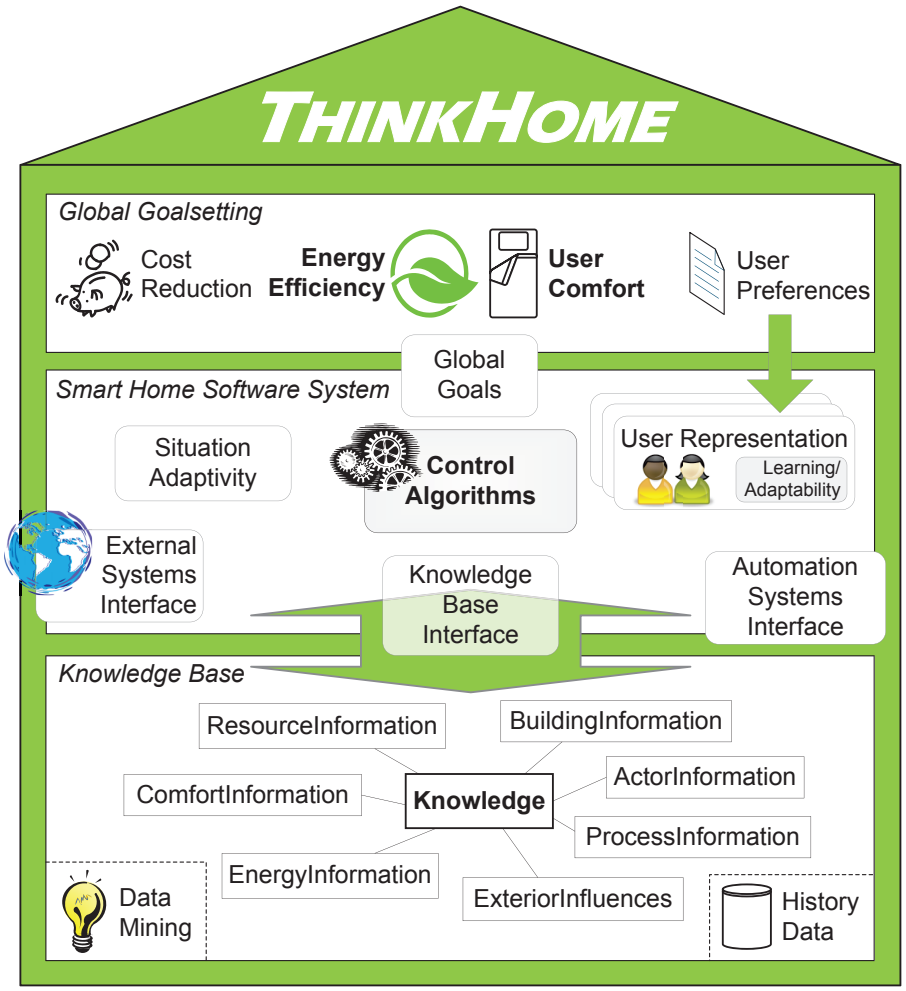

Figure 2.2: Overview of the ThinkHome system architecture

and computation time) for one control task may exist. It is the task of the software system to dynamically select appropriate algorithms during system operation. As smart home control is defined to exhibit intelligent behavior, many control algorithms will make use of mechanisms known from artificial intelligence. The control strategy selection process as well as the execution of the algorithms is supported by several other components that are specified around this core functionality. These other parts are not only needed to support or optimize the control strategies as such but also enable their execution in the environment.

The control strategies operate under the regime of the global goals, i.e. they consider the global goals as parameters in their algorithms and the selection of the

control strategies is influenced by the global goals, respectively. Further input parameters of the control algorithms can be obtained from the extensive knowledge base and from the other smart home control system parts. The *user* part is used to represent the user in the smart home system. As mentioned there is one user software part for each smart home system user[4]. The user part is needed to advocate the users' preferences in the smart home system and in particular concerning the execution parameters of the control algorithms. The user software parts can hence be understood as (software) avatars of the users that take over routine tasks from them and in general take some of the control burden from the users. Within the user part, preferences concerning personal comfort values and those related to energy efficiency are represented. Furthermore, particular special preferences concerning the smart home control, for example specific exceptions, such as windows that shall not be opened when this user is present, can be represented here. Additionally, the user part manages tasks, such as the personalized visualization of energy consumption that operate in close cooperation with the user so that, for example, tailored interfaces for each user may be created. The user part hence also includes the functionality of a user interface. The *user interface*, as its name implies, handles all interaction with the system users. The user interface is unique in the system in the sense that it is intended to be exclusively used by humans (in contrast to the other interfaces that are mainly used by machines, i.e., for machine-to-machine communication). Therefore, the user interface has very special design requirements so that it can be used successfully by humans. Not last, the user interface is the only part of the system that is actively exhibited to the user. Therefore, it represents one main acceptance and success criterion of the whole smart home system.

The *learning/adapability* part is used to monitor the system operation and draw conclusions from this execution. With the help of mechanisms from the artificial intelligence domain, such as context awareness, the system is empowered to adapt itself to changed requirements or changes in the user behavior. For example, manual interventions of users can be identified and incorporated into the control strategies. Similar to this, learning algorithms can be employed to keep track with changing user preferences (i.e., the user is not required to enter a new comfort temperature, but the system derives this temperature from the user's interaction with the system).

Situation adaptivity goes along closely with the availability of a variety of control algorithms. Software can be used to support the system in choosing among different

[4]The term smart home "user" is used intentionally instead of the word "resident" since the system home control system shall also be able to cater for the preferences of house guests. In practice, guests will be offered limited and tailored control functions only due to security and privacy reasons.

algorithms that have the same final control goal. Thus, action alternatives become possible and the system is more likely to operate well in a dynamic environment.

Apart from its control functionality, the smart home control system layer also has five main interfaces that are used for interaction with other components, systems and the environment. First of all, a *global goals interface* is specified. Over this interface, system wide parameters concerning energy efficiency and comfort (in contrast to user specific ones which are handled by the user part) can be accessed in the control part. Therefore, global values need to be defined only at one place in the system but can be used by all control strategies. Also general energy conservation policies or general comfort strategies are provided via this interface. The centralized approach facilities ensures the system wide availability of the parameters and allows to make changes to these parameters easier at a later point in time.

In the direction from the software system to the outside world, the user interface is used for system visualizations, to deliver feedback to the user or to provide general information to the user. The latter aspect includes the illustration of energy consumption data to increase awareness and possibly even induce a behavior change if the feedback is delivered in an appropriate way (cf. "persuasive technologies" as described in [54, 55]). The user interface also functions as the main entry point of users into the system (corresponding to the direction from the outside world/user into the system). Initial configurations as well as parameters and preferences can be entered or adjusted via the user interface. During operation of the smart home system, alarms and feedback are delivered via the user interface. If user input upon these situations is required, it is also gathered via the interface.

The *automation systems interface* establishes the connection to the underlying automation systems of the smart home. While the software layer operates independently from any specific technologies, it is still the case that dedicated protocols enable the automation functionalities in the smart home. The automation system interface hence provides a two-way access to the devices of the automation system, i.e., sensors, actuators and controllers. Via the interface, the values (e.g., set points) computed by the control strategies can be communicated to the automation systems that take over the interaction with the environment and its physical process. Likewise, data from the environment can be captured using sensors. In order to use it within the smart home control system, all data has to pass through the automation system interface.

The smart home control system is also the entry point for external systems or applications (i.e., systems not being part of the core functionality of the smart home

system as presented in the section). The only access point of these external systems is the *external systems interface*. One reason for this is that it becomes possible to implement privacy and security concepts within the gateway software part that implements the interface functionality. Again, information flow via this interface is bi-directional. On one hand, it acts as a point of "entry" for other systems and applications, such as the smart grid, demand side management, performance contracting and similar. Furthermore, the interface can be considered as the gateway to other systems that provide relevant services or relevant information, such as a weather service. In particular, the information providing services are of high importance as they provide an almost inexhaustible source of information. Today, this mainly concerns the vast information available on the Internet, where almost every conceivable kind of information is provided via Web services (weather data, additional product/device information, etc.). On the other hand, the external systems interface is also used by the smart home control software to deliver data to other external information sinks. These can be, for example, power supply companies (that e.g., get information about planned/projected energy consumption) or other smart homes or smart city "inhabitants" (that e.g., receive announcements about current excess energy that could be available from the smart home).

The *knowledge base interface* provides access to all information that is stored in the knowledge base. An overview of the data stored in the knowledge base is provided in the next section. The interface itself can once again be used from the software system side to retrieve data that is needed (mainly) for the operation of the control strategies. Furthermore, the software layer can push data to the knowledge base in order to add it as a new item there or to update some existing piece of information. This "write" access of the knowledge base concerns both data that is generated by the software layer, for example, by means of data analysis and data that originates from other sources (the automation systems or any external information provider).

Obviously, there are more or less strong dependencies among the identified components and these components also need to exchange information or cooperate to successfully master some tasks. For example, control strategies are likely to require input from the learning component to be informed of current changes. It is the task of the implementation to find a suitable implementation/software technology that either provides the necessary communication means between the system components or that realizes the system functionalities as one monolithic program thus eliminating the need for information exchange between components.

2.3.2 Knowledge Base

The knowledge base is an integral part of the overall smart home system. As such, it supports the successful operation of the smart home system and in particular is an enabler of "intelligent" control within the smart home control system. The major contribution of the knowledge base is that it enables the definition and execution of novel control strategies that rely on additional data that would not be available without such a knowledge base. Thus, the knowledge base is designed as a component that can store knowledge and provide it to other (software) systems via a designated interface. Related to the latter requirement, it must be guaranteed by the knowledge base that data is stored in such a way that it can be accessed by machines. This includes communication in both directions, i.e., machines must be able to read data as well as to store data. Additionally, the possibility to manipulate stored data must be provided.

Finally, a knowledge base of a smart home can comprise more than a simple data store. In particular, it could store the relevant data of a smart home already in an intelligent way, for example by including relationships between data items in the representation. Also, a structured approach of storing data could enhance the smart home operation. Following this way, a knowledge base could provide meta information on the stored data, for example a hierarchical structuring of data could already provide the information that a dimming device of technology A and a dimming device of technology B provide the same functionality but just use different communication protocols. Although this is not considered a mandatory requirement of a smart home system, such an intelligence could also support (automatic) classification of information. In knowledge engineering and logics, such a behavior is termed "inference". Inference denotes operations on stored data that have the goal to derive conclusions from these data by using mechanisms that are applicable to description logics.

Hidden relationships between data can be made visible and hence be used to enhance the smart home control strategies. The advantage is that not all relationships need to be modeled explicitly and thus the complexity of the knowledge base is reduced. Furthermore, maintenance of the data as well as guaranteeing consistency of all represented data are facilitated when offering inference mechanisms. Nevertheless, it is the case that the use of inference mechanisms partly limits the flexibility of the data representation. In order to exploit the potential of inference, it is relevant how the knowledge representation is designed and how data is organized within it. In practice, if inference is targeted, the design of the knowledge base should follow

certain design guidelines already from the beginning of its conception. Therefore, the requirements clearly influence the choice of an implementation technology for the knowledge base.

Within the knowledge base, all information that is relevant concerning the operation of a smart home shall be stored. The data shall provide a comprehensive view of the world, more exactly of the world that is visible and relevant to a smart home.

Besides the building structure itself and every device/component that exists in the building, this concerns also users, external systems and moreover parameters of the environment. Since novel control strategies are a main requirement for smart homes, also previously unconsidered information has to be included in the knowledge base. Conceptually, the knowledge base is partitioned into several parts. Each part represents information belonging to particular aspects in the smart home. Similar to the conceptual design of the smart home control system, also the definition of several components of the knowledge base has to be regarded on a conceptual level. Concrete implementations, depending on their technology, may merge components or even further partition the knowledge into further domains.

Figure 2.2 shows the seven main information parts that together form the available knowledge. First of all, the storage of building information is of great importance. The *buildingInformation* part stores information, such as building characteristics that can support optimized control strategies that strive for an energy-efficient operation of the smart home. The information includes the whole building structure with walls, doors, windows, ceilings and roofs with their physical properties (thickness, size, room volume, etc.) as well as concepts, such as rooms and floors. Furthermore, the orientation of the building and its elements are included. Regarding the later use as input parameters for the smart home control strategies, also properties of these elements that come from and are used in the building physics domain, such as thermal conductivities or insulation values are of high relevance. In combination, the required information concerning the building quite closely corresponds to the information that is accumulated during the design and construction phase of a building. The contents of the building information part of the knowledge base are thus similar to the data collected in a Building Information Model.

Apart from concepts relating to the building, also information about the users of the system has to be considered. This *actorInformation* concerns both users in terms of human users (such as inhabitants and guests) but also information on other (software) systems that act as users within the smart home. An example for the latter case are applications such as demand side management that may (autonomously)

execute actions in the smart home. For human users, the knowledge base knows different characteristics (e.g., age and gender) and more importantly keeps user profiles. In a user profile, the preferences of one user (for example regarding comfort but also unwanted actions) are stored.

The elementary operations that describe the users' activities are stored in the *processInformation* part. These are needed to have user, usage and habit profiles available in the system. Furthermore, also information on the basic system processes is kept here. This includes the storage of system actions (processes) that can be executed in the environment to reach a particular world state. For example, the knowledge base includes information on all processes that can be used to cool down a room. In practice it could store the processes "open window" and "activate air-conditioning" that are generally applicable when the temperature shall be reduced.

In the *exteriorInfluencesInformation* part data that describes the environment of the smart home is stored. This part contains foremost concepts that describe information related to the weather as well as outdoor climate information. The weather and climate data can be used to select control strategies and to perform control tasks more energy efficiently.

The main information that is kept in the *energyInformation* component concerns a description of different energy providers and their trading conditions. It also includes information on the kind of energy that is supplied (electricity, thermal energy, gas) and represents the production of this energy (nuclear, gas, water power, etc.). The energy information is especially valuable when envisioning the integration of the smart home into a smart grid, where the availability of this information allows to select the momentarily best option for energy consumption or recovery. This knowledge base part also keeps energy schedules for different occupancy states and scenarios (e.g., day, night, weekends, holidays). In general, the energy information can be used to optimize the energy consumption, for example, by anticipating consumption peaks.

In the smart home system, also information on comfort aspects is of relevance regarding the control approaches. Therefore, a *comfortInformation* component is specified that holds various aggregations of elementary measurement units (e.g., temperature, humidity, luminosity) that provide a notion of comfort to the system (e.g., thermal comfort, visual comfort).

The *resourceInformation* part relates to the actions that can be executed by and in the underlying automation systems. It provides information on the automation services that are installed in the smart home as well as on the equipment that is avail-

able. The resource information part includes household appliances (white goods), consumer electronics (brown goods), and automation devices from the domains of lighting, shading as well as heating, ventilation, and air conditioning (domotics). As the automation networks that host these devices can be of different types, protocols, and manufacturers, also information on them is represented. This knowledge allows to generalize on the devices (i.e., representing them in a generic way), which in turn supports the transparent integration and communication across the different networks. In addition, energy producing components like solar collectors or a thermal heat pump are stored in this part. Hence, a complete model of the energy consuming and producing device landscape in the smart home is stored.

Apart from the components that represent the information kept in the system, the knowledge base also features two components that offer services on the knowledge that is stored within it. First, not only current data will be required for smart home operation but also *history data* is of high relevance. The respective component shall hence ensure that mechanisms are provided that allow to collect and store selected data on a regular basis. The history data component must also provide services that allow to access the stored data from the smart home control part. Second, a *data mining* service is included in the knowledge base layer. As mentioned before, this component can be used to operate on data, for example, with the goal to make "hidden" information visible through logic operations. Furthermore, data mining mechanisms can be used to find patterns within the stored data or to organize the data in a particular way.

Finally, the knowledge base obviously has to share an interface with the smart home control system that it shall support. This knowledge base interface overlaps between the two main system layers and has already been described in the previous section. Regarded from the perspective of the knowledge base, the interface must fulfill the requests from the smart home control part. During an implementation of the interface, attention must be given to ensure the consistency of stored data in case of updates or newly added data.

Taken together, all knowledge parts form the comprehensive knowledge base that supports the smart home control system in its operation. It provides control strategies with a view of the world and supplies the necessary parameters to them. The knowledge part is thus fundamental for the successful and optimized system operation and grounds the smart home system.

Although both layers of the smart home concept presented here are basically independent from each other, an alignment of KB and control system is required to

maximize the benefit in control. Data that is needed by control strategies should ideally be provided by the knowledge base. Decision processes in the smart home control system can be supported by the knowledge base only if the decision criteria are known in advance and mapped to. Since not all control strategies may be known at the design time of the knowledge base, it must not be considered exclusively as static representation. Rather, basic structures will be quite static, while other information will be added only at a later point in time and furthermore change dynamically.

Chapter 3

System Specification

Preface

The smart home system called "ThinkHome" was initially developed as part of an Austrian research project of the same name. The basic research project was conducted at Vienna University of Technology, Automation System Group from 2009 until 2012 under the lead of Prof. Wolfgang Kastner. It was funded by the Austrian research promotion agency FFG under the contract P822 170.

The work within the project was separated into three main parts: the design of a comprehensive software framework, the conception and design of a knowledge base and the development of novel control strategies for the smart home. Due to the complexity of each part, three PhD students (Mario Kofler, Felix Iglesias Vazquez, and myself) were each responsible for one respective part. Although the actual work was done separately, research was conducted as a team. Therefore, the overall concept as presented in Section 2.3 is a result of the joint effort of the whole team. Furthermore, overlaps of the individual works were not only tolerated but actually desired to ensure the development of a functional system. These common parts mainly concern the shared interfaces as well as fundamental concepts such as the basic use cases that were developed jointly to be used as common starting point. However, unless explicitly marked, the work described in this dissertation was accomplished by myself.

In particular, this dissertation deals with the methodologically sound design of the software framework that supports the operation of next-generation smart homes. The software framework interfaces with the comprehensive smart home knowledge base and hosts the control strategies of the smart home.

3.1 Use Cases

This section provides a list of use cases that shall collect the base requirements of a future smart home and in particular for the smart home system part of ThinkHome. The use cases therefore capture different parts of the system, trying to cover as many functionalities as possible. Furthermore, the (desired) system execution paths, related error conditions and failure recovery procedures are modeled.

The methodology to obtain the use cases, the representation of their formal structure and design of the actual use case description are very much aligned with the guidelines provided by Cockburn in his book "Writing Effective Use Cases" [56]. There, Cockburn in short describes the following basic elements of a successful use case. The elements printed in bold face denote the minimum information that is required for the description of a use case. This way of representation is also known as *Cockburn style*.

Use case number and name The name of a use case describes the goal that is connected to it. Additionally, a number is often provided for easier referencing to the use case.

Context of use The goal of the use case is described in more detail. This element can also contain information on the conditions when the use case normally occurs.

Scope The scope is used to assign this use case to a particular system. It describes the boundaries between the system that is specified through the use case and possibly existing other (sub)systems and hence also captures the (spatial) validity of the use cases.

Goal-level Three goal levels exist: user-goal (also called "blue" goals), strategic ("white") and subfunction ("indigo" or "black") level goals. The user-goal level is of highest importance. It is used to characterize goals that represent the basic and most important tasks in the system. Strategic goals describe the context in which the system operates. Their specification helps to view the system on a large scale, e.g. a company level. Finally, indigo goals can be seen as subgoals of a blue goal. They need to be achieved so that also the user-goal succeeds.

Primary actor Cockburn describes an actor to be "anything that has a behavior". It can be a system, a person, a computer program, a software or company. The

primary actor of a use case is "the external actor whose goal the use case is trying to satisfy, the actor whose action initiates the use case activity". For each use case there is always only one dedicated primary actor, which must furthermore be different from the system under discussion (SuD). The latter requirement ensures that the described system remains a black box.

Secondary actor The secondary actor is an external actor that provides some service to the system under design. Their identification is useful to capture all the interfaces the SuD has.

Stakeholders and interests As the name implies, this item captures (additional) stakeholders that have an interest or play a role in the use case. The primary and secondary actors are also part of the stakeholders but are modeled standalone due to their importance for any use case.

Pre-condition The pre-condition can be any condition that must be fulfilled or guaranteed before the use case can be executed. It is mostly written as assertion to some world state.

Success end condition The success end condition describes the state of the world when the goal is successfully completed. In other words, it states "what interests of the stakeholders have been satisfied at the end of the use case".

Failed end condition The failed end condition describes the state of the world when the goal could not be successfully completed or is abandoned.

Failure protection condition This item shall ensure that failures in reaching a goal do not cause damage to a system. It may for example contain information how a safe system state can be reached if a goal has to be abandoned/is failed.

Trigger The trigger describes an event that initially starts a use case. It could be a timer or an interrupt.

Main success scenario The main success scenario is composed of an ordered collection of actions that are used to achieve some goals. It shall describe the most likely or common sequence of actions so that the goal of the use case is achieved in the system. Therefore, each action step is numbered and described shortly. There may be more than one possible sequence of actions to achieve the goal, however, there is always only one main success scenario per use case.

Extensions Extensions describe alternative scenarios that can occur when the use case is executed and hence model alternative execution paths. Extensions always refer to action steps of the main success scenario and provide conditions under which the alternate path of execution is taken. They can thus be compared with IF statements in programming languages.

It is important to note that there might by further system functionalities in the final system that are not covered by the use cases presented in this chapter. Furthermore, the use cases in this dissertation represent the state and considerations that were made at the conceptual system design phase, i.e., at the beginning of the system design and in particular *before* the system has actually been put into practice. This decision is justified by the inherent understanding what the use cases can and should contribute to the system engineering process. In this dissertation, the use cases are primarily regarded as a valuable means to identify the requirements of the future system. Chronologically seen, they thus have to be defined at the beginning of the system development (i.e., during the system design phase) so that they can be referenced and used as guidelines during later system development and system evaluation. This view is clearly different from the understanding that use cases can also be used as a posterior formalization or documentation of an already practically realized system.

Another important point is the question whether use cases should be adapted continuously during the whole system engineering life cycle or not. In practice, three different approaches exist. First, the use cases can be adapted iteratively. This has the benefit that there is always a current requirements description but comes at the price that the initial intentions may no longer be visible or might at least be blurred. Second, the use cases can be captured once and not be changed afterwards (except for the elimination of severe design flaws). This approach is adopted in this dissertation. Third, Cockburn also mentions the possibility that use cases are written *after* the system has been implemented. In this case the use cases provide a kind of documentation of the actually built system, which can be a reasonable approach for example when later modifications are planned. However, in the latter case, the use cases are obviously not usable during the system engineering process and can thus not contribute to a better system design.

The motivation to permanently leave the use cases in the form as they were initially specified during the design is to reflect the system engineering process that was conducted during the dissertation work. In particular, it shall allow for a comparison of the initial requirements with the later design decisions and solutions. It shall also

give the reader the opportunity to better review the evolution of the system from its use cases over further design artifacts towards a complete system specification and prototype implementation.

For the smart home, several use cases were identified that specify and illustrate on one hand the global goals of the overall system. These system-wide functions are described by means of two "strategic" level use cases. On the other hand, more concrete operations and functionalities of the smart home are captured in seven further use cases. The latter use cases are of the "user-goal" type and relate in particular to the tasks that are executed in the smart home control part of the overall system. Furthermore, they illustrate the interactions between the system parts and also with external systems and services.

3.1.1 Use Case: Energy Efficiency in the Smart Home

USE CASE: S01 - Assure Energy Efficiency Operation of Services/Devices in the Smart Home

CONTEXT OF USE:

The smart home realizes the optimized operation of energy facilities in the home. It coordinates the use of the energy consuming devices and processes, chooses among alternative processes and takes advantage of information stored in the knowledge base to reduce energy expenditure.

SCOPE: Smart home

GOAL-LEVEL: Strategic

PRE-CONDITION: –

SUCCESS END CONDITION: Energy consumption in the smart home is minimized

FAILED END CONDITIONS:
- Existing energy savings potentials remain unused
- Energy consumption in the smart home is increased

FAILURE PROTECTION CONDITION: Manual operation of the building services is possible

PRIMARY ACTOR: Smart Home (SH)

ACTORS:
- Knowledge base (KB)
- Smart Home System (SHS)
- Automation system (AS)

STAKEHOLDER: Inhabitant (IH)

TRIGGER: – (permanent goal)

MAIN SUCCESS SCENARIO: The smart home is aware of all building services and devices. It schedules and supervises the operation of the devices and building services in a way so that the energy consumption for the building operation is minimized. The smart home knows different control strategies and executes them under consideration of the current world state. It is aware of energy providers and renewable energy sources and integrates this information in its control strategies. The smart home has control over all building services and can influence their status and operation.

EXTENSIONS:
- The smart home knows the occupancy schedules or user habits of the inhabitants. It adapts its control accordingly.

3.1.2 Use Case: Increased Comfort in the Smart Home

USE CASE: S02 - Assure Comfort-oriented Operation of Services/Devices in the Smart Home

CONTEXT OF USE:

The smart home ensures that the user comfort is maximized during the operation of facilities in the home. It coordinates device operations and the execution of services under permanent comfort consideration. It ensures that user comfort parameters are kept in the smart home and that comfort related operations are provided with the corresponding relevant parameter inputs.

SCOPE: Smart home

GOAL-LEVEL: Strategic

PRE-CONDITION: –

SUCCESS END CONDITION: User comfort in the smart home is maximized

FAILED END CONDITIONS:
- Existing user comfort optimizations remain unconsidered
- User comfort in the smart home is decreased

FAILURE PROTECTION CONDITION: Manual operation of the building services is possible

PRIMARY ACTOR: Smart Home (SH)

ACTORS:
- Knowledge base (KB)
- Smart Home System (SHS)
- Automation system (AS)

STAKEHOLDER: Inhabitant (IH)

TRIGGER: – (permanent goal)

MAIN SUCCESS SCENARIO: The smart home is aware of all building services and devices. It schedules and supervises the operation of the devices and building services in a way so that the comfort preferences of the inhabitants are preserved. The smart home knows different control strategies and executes them under consideration of the current world state. It is aware of the occupancy and an occupancy schedule and integrates this information in its control strategies. The smart home has control over all building services and can influence their status and operation.

EXTENSIONS:
- The smart home detects when a new device/facility is added to the system.
- The smart home accesses its knowledge base for further information on devices, inhabitants and processes
- Not all building services are automated or networked. The smart home takes control over a limited set of services only.

3.1.3 Use Case: Air Quality

USE CASE: TH01 - Assure Air Quality

CONTEXT OF USE: The smart home system (SHS) automatically preserves the air-quality in a building. It periodically receives the current air quality from the automation system and validates it against reference values derived from common standards (e.g., the American Society of Heating, Refrigerating and Air-Conditioning Engineers (ASHRAE) [57]). When air quality drops below health or comfort standards, the SHS queries the knowledge base (KB) to retrieve possibilities to reach air quality. From this variety of facilities and processes, the SHS chooses the most energy efficient process. The SHS must ensure that the selected process is executable at the moment concerning the current internal and external world states in the home. After choosing the best alternative, the smart home system takes care of the execution of the actions belonging to the selected process.

SCOPE: Smart home system/knowledge base subsystem

GOAL-LEVEL: User-goal

PRE-CONDITION: Air quality is below the recommended range

SUCCESS END CONDITION: Air quality is within the recommended range

FAILED END CONDITION: Air quality remains unchanged

FAILURE PROTECTION CONDITIONS:
- Air quality control switched to manual operation
- Windows and doors are in a secured state (security requirement)
- Windows and doors are in a safe state (safety requirement)

PRIMARY ACTOR: Smart Home System (SHS)

ACTORS:
- Knowledge base (KB)
- Automation system (AS)

STAKEHOLDER: Inhabitant (IH)

TRIGGERS:
- Time
- Automation system reports air quality out of recommended range

MAIN SUCCESS SCENARIO:
The SHS actively retrieves the current air quality data from the sensors of

the automation system. The data is stored in the KB. If the air quality is within the desired range, no action is taken. If the air quality is not within the recommended range (concerning guidelines, preferences), the SHS requests air quality improvement options from the KB. The KB returns a list of currently applicable processes to increase air quality. From this list, the SHS selects the most appropriate one – given the SHS's current knowledge on the state of the environment – for execution. The SHS initiates the air quality improvement operations and monitors their execution via the automation system.

EXTENSIONS:
- The air quality measurement data can be pushed to the SHS in case of thresholds being exceeded. In this case or when air quality is detected to be health critical, immediate action is taken without requesting support from the KB.
- No data is returned from the KB. The SHS executes standard actions.
- The KB returns that currently no applicable processes to increase air quality exist. The SHS executes standard actions.
- Several facilities exist in the home that can be used for air quality improvement (e.g., ventilation equipment or window openers). The SHS can request further data on these facilities from the KB to support the decision process.
- After execution of actions, air quality is still outside the desired range. The SHS selects another improvement option and executes it.

3.1.4 Use Case: Thermal Comfort

USE CASE: TH02 - Assure Thermal Comfort

CONTEXT OF USE: The smart home system permanently and automatically aims at ensuring thermal comfort. Comfort conditions are based on the actual and scheduled occupancy and the current state of the home or a specific room. For certain control approaches, knowledge of time parameters (thermal inertia), weather conditions and building structure is needed. The Smart Home System (SHS) periodically monitors the current temperature of different parts of a home from the automation system. Under consideration of current and expected future occupancy and related preferences of occupants, it initiates actions that influence the temperature in a room. In particular, the SHS selects the most appropriate control possibility among a set of different heating/cooling options, decides on the most appropriate point of time to start the selected

process and takes care of the execution of the actions belonging to the selected process.

SCOPE: Smart home system/knowledge base subsystem

GOAL-LEVEL: User-goal

PRE-CONDITIONS:
- Occupancy (or expected occupancy)
- Air temperature below or above comfort temperature

SUCCESS END CONDITION: Temperature is within preferred range

FAILED END CONDITION: Temperature remains unchanged

FAILURE PROTECTION CONDITIONS:
- Temperature control switched to manual operation
- Setback temperature is provided

PRIMARY ACTOR: Smart Home System (SHS)

ACTORS:
- Knowledge base (KB)
- Automation system (AS)

STAKEHOLDER: Inhabitant (IH)

TRIGGERS:
- AS reports temperature outside desired range
- Occupancy change
- Time (Scheduled/predicted occupancy)

MAIN SUCCESS SCENARIO:
The SHS actively monitors current and expected future occupancy of a room. It also periodically retrieves the current temperature values of a room from the sensors in the automation system. For an occupied room, the SHS retrieves the thermal comfort preferences of all present persons from the KB. If the temperature is within the desired comfort interval, no action is taken. If the temperature is not within the desired range, the SHS requests temperature control options from the KB. The KB returns a list of currently applicable processes to control room temperature. From this list, the SHS selects the

most appropriate one – given the SHS's current knowledge on the state of the environment – for execution. The SHS initiates the temperature control actions and monitors their execution via the automation system.

EXTENSIONS:
- The temperature measurement data is pushed to the SHS in case of thresholds being considerably exceeded. In this case or when temperature is detected to be health critical, immediate action is taken without requesting support from the KB.
- Several facilities (e.g., a conventional heating system, geothermal heat pump) exist in the home that can be used for temperature control. The SHS can request further data on these facilities from the KB to support the decision process.
- At the moment no presence is given, but occupancy is expected in the near future. The SHS retrieves the comfort preferences of the expected occupants from the KB.
- If no occupancy is given at the moment, but occupancy is expected with the near future, the starting time of temperature control actions to reach comfort values until the point of occupancy is calculated.
- If occupancy changes from occupied to non-occupied status, the corresponding set point temperature value is updated to the defined set back temperature.
- After execution of actions, the temperature is still outside the desired range. The SHS selects another temperature control option and executes it.

3.1.5 Use Case: Visual Comfort

USE CASE: TH03 - Assure Visual Comfort

CONTEXT OF USE: The smart home system permanently and automatically aims at ensuring visual comfort. When rooms are occupied, the SHS compares visual comfort preferences against the current data provided by the automation system and initiates adjustments to lighting/shading equipment if required. When rooms are unoccupied, minimum energy consumption of the lighting system is ensured by the SHS. The SHS can obtain further information regarding applicable processes and available facilities from the knowledge base (KB). To establish visual comfort, the SHS chooses the most energy efficient control option and takes care of the execution of the actions belonging to the selected process via the automation system (AS).

SCOPE: Smart home system/knowledge base subsystem

GOAL-LEVEL: User-goal

PRE-CONDITIONS:
- Occupancy
- Visual conditions out of comfort bounds

SUCCESS END CONDITION: Visual comfort established

FAILED END CONDITION: Visual conditions remain unchanged

FAILURE PROTECTION CONDITION: Lighting/shading control switched to manual operation

PRIMARY ACTOR: Smart Home System (SHS)

ACTORS:
- Knowledge base (KB)
- Automation system (AS)

STAKEHOLDER: Inhabitant (IH)

TRIGGERS:
- AS reports visual comfort conditions outside of desired range
- Occupancy
- Time (Scheduled/predicted occupancy)

MAIN SUCCESS SCENARIO:
The SHS periodically assesses the lighting situation in a room via data provided from the AS. If the visual comfort conditions are within the desired comfort interval, no action is taken. If the SHS detects that visual comfort is outside the comfort zone, it requests operations to influence visual comfort from the KB. The KB returns a list of devices and operations that influence the visual situation in a room (e.g., blind positions, artificial light sources). From this list, the SHS selects the most appropriate one – given the SHS's current knowledge on the state of the environment – for execution. The SHS initiates the lighting control actions and monitors their execution via the automation system.

EXTENSIONS:
- The SHS adapts visual comfort according to a given visual comfort schedule.

- The lighting situation data is pushed to the SHS in case of thresholds being considerably exceeded. In this case, immediate action is taken without requesting support from the KB.
- Several facilities exist in the home that can be used to influence visual comfort. The SHS can request further data on these facilities (e.g., operation modes, control options) from the KB to support the decision process.
- If occupancy changes from occupied to non-occupied status, the corresponding visual comfort goals are adapted to this new situation.
- After execution of actions, visual comfort conditions are still not reached. The SHS selects another lighting control option and executes it.

3.1.6 Use Case: Efficient Operation of Energy Consumers

USE CASE: TH04 - Assure Energy Optimized Operation of Household Appliances and Entertainment Devices

CONTEXT OF USE:

The smart home system optimizes energy consumption of household appliances and entertainment devices when they are in an idle state. When a room is unoccupied, the system identifies energy consuming devices, such as coffee machines, TVs, radios, etc. that are currently in an idle state. The SHS may also verify the occupancy schedule of the room in question. When no occupancy is detected in the near future, the system initiates actions to disconnect selected equipment from the power supply to save energy. It takes care of the execution of the actions belonging to the selected process via the automation system (AS).

SCOPE: Smart home system/knowledge base subsystem

GOAL-LEVEL: User-goal

PRE-CONDITION: Energy consumers are available

SUCCESS END CONDITION: Energy consuming devices do not waste energy when unused

FAILED END CONDITION: Energy consumption remains unchanged

FAILURE PROTECTION CONDITIONS:
- Devices can be controlled manually
- Devices stay operational

PRIMARY ACTOR: Smart Home System (SHS)

ACTORS:
- Knowledge base (KB)
- Automation system (AS)

STAKEHOLDER: Inhabitant (IH)

TRIGGERS:
- Occupancy
- Time (Scheduled/predicted occupancy)

MAIN SUCCESS SCENARIO:
The SHS periodically monitors the occupancy in a room. When no occupancy is detected in a room, the SHS determines if the room is expected to be occupied in the next X hours. If no occupancy is expected, it queries the KB for all energy consuming devices in the room that are in an "idle" state. The KB returns a list of these devices. The SHS checks for each device if it needs permanent power supply and marks all devices that can be removed safely from power supply. The SHS initiates the actions to separate the selected devices from the power grid and monitors the execution of the actions via the automation system.

EXTENSIONS:
- No energy consuming devices are installed. No action is taken.
- Changes in occupancy are pushed to the SHS by the AS.
- The SHS adapts device states according to a pre-defined schedule (user profile).
- Occupancy is expected within the next X hours. No action is taken or already taken actions are reversed, respectively.
- The SHS queries the KB for further information (device states, usage profiles) on a device.
- Several devices are connected to the same socket. The SHS marks these devices and informs the inhabitant of possible actions.

3.1.7 Use Case: Efficient Integration of Local Energy Producers

USE CASE: TH05 - Assure the Efficient Integration of Local Energy Producers

CONTEXT OF USE:

The smart home system controls electric consumers in the smart home to use locally produced energy in the best way. The scheduling process includes weather information and required device operations to calculate when enough renewable energy will be produced and schedules energy-intensive processes accordingly.

SCOPE: Smart home system/knowledge base subsystem

GOAL-LEVEL: User-goal

PRE-CONDITIONS:
- Energy producers are available
- Energy is produced in usable quantities

SUCCESS END CONDITION: Locally produced energy is used in the smart home for (scheduled) tasks

FAILED END CONDITION: Locally produced energy is not used in the smart home but fed back to the electricity grid

FAILURE PROTECTION CONDITION: Devices can be operated using energy from the electric grid

PRIMARY ACTOR: Smart Home System (SHS)

ACTORS:
- Knowledge base (KB)
- Automation system (AS)
- Weather service (WS)

STAKEHOLDER: Inhabitant (IH)

TRIGGERS:
- Weather forecast
- Energy intensive operation scheduled

MAIN SUCCESS SCENARIO:
The SHS retrieves a list of local, renewable energy producers from the knowledge base and a weather report from a weather service. The SHS periodically monitors the weather predictions and calculates the predicted amounts of energy that will be produced in the future. It matches these predictions with a list of scheduled device operations and calculates an operation schedule for

these devices, where renewable energy is prioritized. The SHS initiates the actions of the selected devices with respect to this schedule and monitors the action execution via the automation system.

EXTENSIONS:
- The SHS receives weather data from a local weather station.
- The SHS does not have any weather information. No action is taken.
- The SHS queries the KB for further information on a local producing device (e.g., power output, device states) or on an energy consuming device (e.g., energy consumption, operation modes).
- The SHS cannot find a match of locally produced energy and device operations. Devices are operated using external energy. Locally produced energy is fed to the electricity grid.
- The weather prediction is wrong. The device operations are re-scheduled or external energy is used.

3.1.8 Use Case: Efficient Integration of Energy Providers

USE CASE: TH06 - Assure the Efficient Integration of Energy Providers

CONTEXT OF USE: The smart home system aims at selecting and using energy providers that produce energy in a sustainable way and sell energy at the lowest price. From a range of energy providers, the SHS chooses the provider with the most ecological way of energy production (energy mix) and with respect to the offered energy tariffs. Energy from this provider is then purchased in order to satisfy the energy demand of the controlled building.

SCOPE: Smart home system/knowledge base subsystem

GOAL-LEVEL: User-goal

PRE-CONDITIONS:
- High energy expenditure expected (energy intensive processes are scheduled)
- Local energy production cannot currently satisfy energy demand
- Local energy production is not expected to satisfy energy demand within time-frame

SUCCESS END CONDITION: Energy is bought at best conditions (financial, ecological)

FAILED END CONDITION: No decision can be made on preferable energy provider

FAILURE PROTECTION CONDITION: Energy is obtained from random or standard energy provider

PRIMARY ACTOR: Smart Home System (SHS)

ACTORS:
- Knowledge base (KB)
- Automation system (AS)

STAKEHOLDER: Inhabitant (IH)

TRIGGERS:
- Energy intensive operation scheduled
- Time

MAIN SUCCESS SCENARIO:
The smart home system retrieves a list of external energy providers when an energy intensive operation is scheduled for an execution. It searches this list to find an ecologically friendly energy provider that offers the best selling conditions (prices, tariff). It then schedules the energy intensive operations under consideration of the tariffs offered by the selected energy provider. At the scheduled point in time, the SHS monitors the execution of the initiated actions via the automation system.

EXTENSIONS:
- The SHS retrieves a current list of energy providers at regular intervals.
- The SHS also inspects tariffs and expected prices in the near future.
- No (better) energy provider can be found. Energy is obtained from the default provider.
- The best conditions are outside of the designated time frame for the operation. The SHS checks for shifting possibility and accordingly updates schedule.
- The SHS switches from the default energy provider to a different one, if the selected energy provider for the execution of the energy intensive task is different from the default provider.

3.1.9 Use Case: Preemptive Heating/Cooling

USE CASE: TH07 - Preemptive Operation of HVAC services

CONTEXT OF USE:

The smart home system realizes preemptive heating, cooling and ventilation of the building, where comfort conditions can be provided without requiring notable amounts of energy. For this purpose, environmental conditions are exploited, for example, sunlight may be used to heat a room and cool night air to lower temperature in summer. The system uses weather information (favorable conditions, such as specific weather situations) to calculate an HVAC schedule. It favors the use of local energy or free cooling and free heating to keep or establish comfort values in the home. In order to use "free" HVAC mechanism the system may temporarily overrule comfort parameters within predefined borders (time and range).

SCOPE: Smart home system/knowledge base subsystem

GOAL-LEVEL: User-goal

PRE-CONDITIONS:
- Weather situation and forecast are known
- Occupancy schedule is known
- User preferences and comfort parameters are known

SUCCESS END CONDITION: HVAC goals are reached with least energy effort

FAILED END CONDITIONS:
- HVAC goals are not reached
- Energy effort higher than expected

FAILURE PROTECTION CONDITION: HVAC operations are executed at scheduled points in time

PRIMARY ACTOR: Smart Home System (SHS)

ACTORS:
- Knowledge base (KB)
- Automation system (AS)
- Weather service (WS)

STAKEHOLDER: Inhabitant (IH)

TRIGGERS:
- Weather forecast
- Energy intensive operation scheduled

MAIN SUCCESS SCENARIO:
The SHS requests comfort parameters and an occupancy schedule for the next X hours from the KB. It identifies rooms that will have changes in their HVAC setpoints in the near future. The SHS requests characteristics of the selected room and possible HVAC operations in the room from the KB. It then calculates an HVAC strategy under consideration of current and future weather conditions as well as occupancy where "free" measures are preferred (e.g., night purge). The SHS initiates the actions of the selected devices with respect to this schedule and monitors the execution of the selected actions via the automation system.

EXTENSIONS:
- The SHS monitors the trend of comfort values (e.g., temperature) in a room and initiates actions before thresholds are exceeded.
- The SHS informs the user of possible advantageous actions via the user interface.
- A weather change is detected. The SHS re-evaluates the new situation and aborts unsuited actions.

3.2 Agent Paradigm, Multiagents and Agency Type

The smart home control part needs to be further specified and designed in more detail before it is ready for implementation. On the road to the implementation one important decision concerns the choice of a technology and software paradigm for the system implementation. This decision has a major influence on the design process and thus needs to be made *before* a detailed (i.e., ready-for-implementation) system design specification is developed. Today, many paradigms exist but their differences are often blurred when looked at closer. Furthermore, many overlaps in functions and concepts exist as well as programming languages not always adhere to one programming paradigm exclusively. The advantage at this step is that thanks to the large functional range of programming languages today, the choice needs not ultimately boil down to one selected paradigm, but may combine the advantages of several of them. The reason is that most state-of-the-art programming languages (C++, C#, Java, etc.) today support more than one programming paradigm and allow the mixed use of them in one implementation project.

Still, in a project of the complexity and heterogeneity of ThinkHome, not many paradigms remain as sensitive implementation alternatives after a first closer look.

Functional programming might be an option regarding the knowledge representation and the exposure of this knowledge to other system parts. Also, it might come handy for the specification of control goals, where the goal could be separated from different realization strategies. Nevertheless, for the implementation of a complete smart home system, functional programming clearly misses the representations of changeable data objects and of a state concept.

Scripts could pose an interesting alternative especially regarding the description and execution of control strategies which, in the smart home, are composed more or less of sequences of perceptions, computations, and execution task. However, scripts are a rather rigorous means to handle control strategies as they are prone to stripe most flexibility from them. Additionally, scripting approaches have limited support or offer little convenience for an integration of external services or for the creation and handling of a GUI.

The automata paradigm bears resemblance to the approaches used in programming automation systems. Central concepts are states as well as transitions and transition conditions between these states. Again, automata could be quite well used for the implementation of the control strategies although the system would be largely deprived of its flexibility as uncertainty is not an inherent characteristic of automata. In particular, automata are not designed to operate in very unpredictable environments as all failure conditions and recovery strategies have to be modeled explicitly in the automaton. Additionally, the design and implementation of a flawless, error-free automaton for a system in the dimension to cover a whole smart home is challenging, if not being infeasible at all.

The procedural programming paradigm has many overlaps with the system concept of a smart home that has been developed so far. In particular, it matches the modularity of the smart home system and provides means to encapsulate system functions in separate modules. From an isolated view, procedural programming however has to be considered as no longer being state-of-the-art in programming as it has been superseded by object-oriented programming. In practice, procedural and object-oriented programming go along together very well. Object-oriented adds further concepts and possibilities to the pool of programming functions, but relies on many concepts known from procedural programming. Hence, it is also a serious candidate for the implementation of the smart home system as it offers modularity, abstraction capabilities, message exchange among components and, not last, has become one of the most frequently used programming paradigms today.

Finally, agent-oriented programming offers another alternative for the implementation of the smart home system. Agent-oriented designs are regarded to have object orientation as one of their main ancestors, but were brought up mostly in the artificial intelligence community. Hence, the agent paradigm also incorporates several features that are widely used in artificial intelligence. In the agent case, programming is oriented around the concept of agents which are related to objects but provide some different characteristics. In general, agent-oriented programming shares many aspects, such as modularity and function encapsulation with object-oriented programming. Nevertheless, on one hand it also introduces some constraints that can be beneficial for the system design and, on the other hand agent orientation offers additional functions and aspects for the final smart home system. Thus, the object-oriented and the agent-oriented programming paradigms are identified as (the only) potential programming approaches for the implementation of a comprehensive smart system as outlined in Section 2.3.

In a smart home, control and management tasks have to be performed in a dynamic, non-deterministic and hence complex environment. The environment of a smart home is dynamic because it frequently changes over time. This not only concerns environmental conditions, such as the weather but for example also includes occupancy or the state of external systems. Similar, the environment is largely non-deterministic, not only due to unpredictable weather conditions but also because control over human users or over external services is not in the hands of the smart home software system. These challenges in particular result in the problem that no sensible model of the world can be developed and stored due the complexity of the "smart home" world as such.

Other important aspects concerning the selection of an appropriate programming paradigm are the special characteristics of smart home control. Smart home control functions are commonly distributed across different devices without necessarily featuring a central controller or management instance in the distributed system. This characteristic is quite different from many other software systems where central control instances exist that may also coordinate functions and their execution. In contrast to this, software in a smart homes has to support decentralized operation of subsystems, self-organization of these subsystems and can neither demand nor rely on synchronization among the parts. This necessarily also increases the complexity of the software that shall be used in the smart home. Practice has shown that the effort to design, implement, test and maintain the software of such a distributed sys-

tem is high if not the optimal programming paradigm is chosen. In this context, the agent paradigm is known to especially support design and implementation of systems in which interaction is of particular importance and where system parts have their own thread of control, which is the case in smart home control.

A common reaction to these unconventional non-standard requirements might be to especially follow proven and well-established programming paradigms such as pure object orientation. However, this may have two unwanted side effects. First, other paradigms, such as agent orientation support additional and more tailored functions than object orientation does. Second, the development of most object-oriented programs is accompanied by following standard templates and guidelines (i.e., design patterns in object-oriented programming) which considerable ease implementation and to a certain degree also ensure that adequate functionalities are provided. For the smart home case, neither such templates nor many practical examples exist that cover the entirety of the given programming tasks. This relates again to the heterogeneity of smart homes and smart home systems where control strategies must be coordinated with knowledge bases. While for the latter well-established design patterns exist, no guidelines for a tight combination with the control strategies are available.

Furthermore, some required characteristics of the smart home system are hard to achieve when following object-oriented programming. One example is that control software for a dynamic environment such as the smart home must be able to compensate potential failure conditions (e.g., erroneous or incomplete information on the current state of the environment or the failure of devices and subsystems) without becoming fully dysfunctional. The dynamic environment of a smart home also shows in other uncertainties, such as a spontaneous occupation of rooms. Again, the smart home software must not only be able to tolerate these unexpected world states but must react adequately to them. This requirement is particularly hard to guarantee in an object-oriented system as these systems require their environment to remain static during the execution of their procedures. Obviously, this is almost impossible in the smart home due to many reasons. For example, it cannot be guaranteed that the weather situation is constant during the execution of a heating process.

The ideal smart home system must also be flexible in terms of its functionalities and easy to extend (e.g., to integrate control of further building services). Likewise, later changes or adaptations should be feasible with reasonable effort. When comparing object-oriented and agent-oriented systems in this respect, it can be seen that the agent paradigm prominently advocates loosely coupled software entities as its

basic building blocks. This loose coupling comes to help to master the complexity of a comprehensive smart home system and also facilitates later extension, exchange or maintenance of system functionalities or capabilities.

Finally, one top level requirement of smart homes is to realize improvements and benefits in a home both in terms of comfort and energy efficiency. As outlined before, this is only possible with a great amount of human involvement or by implementing an intelligent system. Such a system should be capable of executing routine (and yet complex) tasks autonomously and on behalf of a user, but without requiring user interaction (cf. Section 2.2). This demand for an intelligent, autonomous system is another clear indicator that points into the direction of using agent-oriented software engineering as paradigm because the related software entities (agents) are characterized by having a goal-directed behavior.

3.2.1 Agent Orientation in the Smart Home

From a computer science point of view, agents can be seen as independent software components that encapsulate some specific functionality. However, agents hold more benefits than being merely a reincarnation of object-oriented distributed systems. Their most significant characterization is that agents as software components exhibit autonomous behavior. Compliant with its name "agent", an agent acts on behalf of somebody or something, and constantly strives to fulfill its "owner's" goals. In further consequence this means that an agent decides for itself if, when and which actions to execute, and explicitly acts without being told what to do neither from a user nor encoded in its implementation. All together these features require at least basic intelligence to be executed reasonably and moreover differentiate agents from objects.

Today, similar to the smart home, no common definition for an agent exists. This is also due to the ambiguous use of the word "agent" in many different domains and research fields, among them software engineering, artificial intelligence, production industry and robotics or also news syndication. The only common denominator is that agents act on behalf of somebody or something. In this dissertation, agents are used as software engineering paradigm and to some extent also as artificial intelligence entity. Thus, a basic definition for these two fields with main emphasis on "software agents" is sought.

A basic definition of agents is given by the Foundation for Intelligent Physical Agents (FIPA). In [58], O'Brien states that "an agent is a piece of software capable of acting intelligently on behalf of a user or users, in order to accomplish a task". Hence, he considers intelligent behavior as a main characteristic of agents. A definition

that rather highlights the software aspects of an agent is given by Shoham in [59]: An agent is "a software entity which functions continuously and autonomously in a particular environment, often inhabited by other agents and processes". However, in the same work he also states that – viewed from an artificial intelligence perspective – an agent can be seen as "an entity whose state is viewed as consisting of mental concepts such as beliefs, capabilities, choices and commitments". Similar to the above definitions and in some way aggregating them, Nwana [60] goes one step further and tries to characterize agents according to the three categories shown in Figure 3.1.

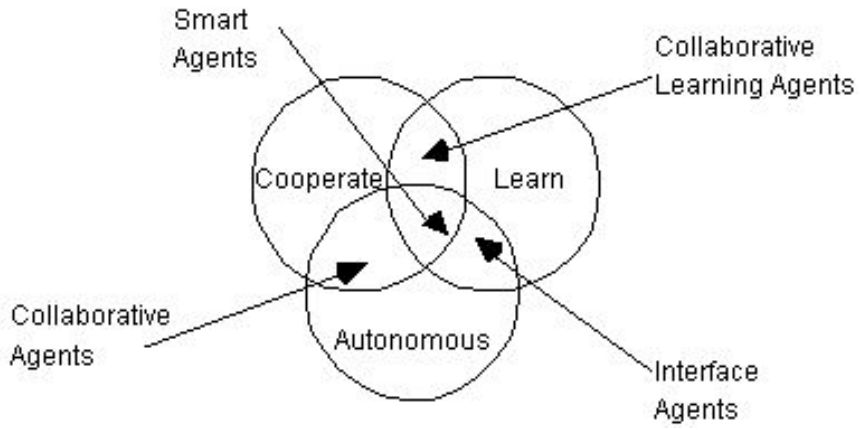

Figure 3.1: Agent topology with a basic classification of agents [60]

This definition of an agent according to the topology of Nwana is also adopted in this dissertation. Thus, an agent is a software entity that exhibits autonomous behavior, is capable of adapting to its environment (i.e., capable of learning) and can interact and cooperate with other software entities. The latter characteristic of cooperating agents leads to the notion of a *multiagent system (MAS)*. As the FIPA in [58] puts it: "Agents, like humans, co-operate so that a society of agents can combine their abilities to resolve problems". Agents can thus cooperate with other agents to solve more complex tasks thereby realizing a type of distributed artificial intelligence.

Another reasonable extension of the agent definition is to speak of *intelligent* agents. These agents which are sometimes also referred to as *rational* agents[1] have clearly defined goals and always work towards these goals by autonomously selecting from a set of different achievement strategies. Furthermore, intelligent agents are ca-

[1]Russell and Norvig [61] define a rational agent as an agent that always selects the action that achieves best results (or that is expected to do so) with respect to the agent goal.

pable of making rational decisions based on the knowledge of their environment that is available to them. As in automation systems, this interaction with the environment is achieved through sensing and actuating components in an agent. According to Russell and Norvig [61], an intelligent agent with the above features is termed *simple reflex agent*.

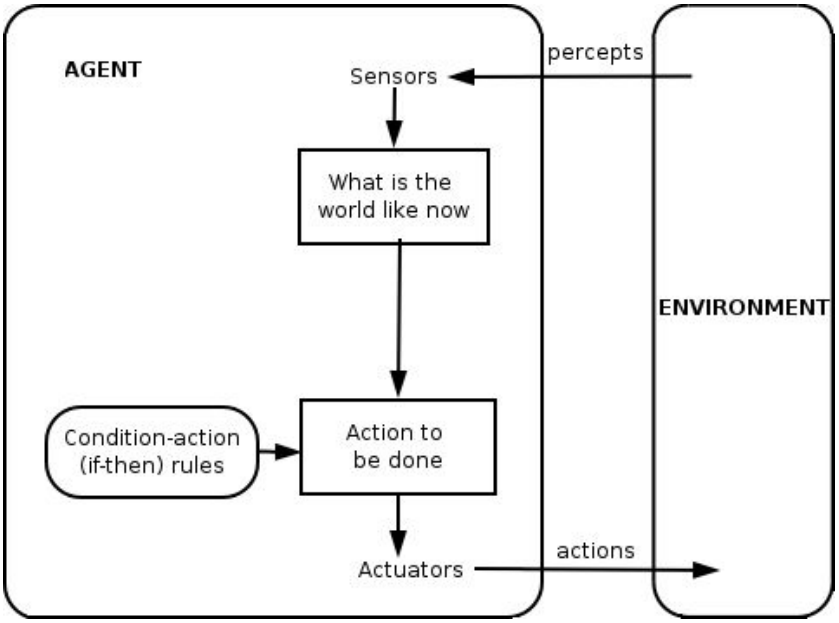

Figure 3.2: Simple reflex agent [61]

Starting from this simple agent model with rather limited intelligence (i.e., if-then rules), agents can have several additional capabilities than autonomy and basic sensing/actuating mechanism that greatly extend the possibilities of an agent and hence make agents more intelligent. Among these capabilities, reactivity, proactiveness and social ability are considered most important. These aspects are described in [62] as follows:

- **Reactivity** Agents have knowledge about their environment and can react to events in the environment.
- **Proactiveness** Agents have goals that they try to achieve during system operation. The agents autonomously act towards these goals without requiring external stimuli.

- **Social ability** Agents can interact and may cooperate with other agents to achieve their goals.

While Russell and Norvig already consider the simple reflex agent as a basic incarnation of an intelligent agent, other sources require a higher degree of intelligence and therefore demand that an agent fulfills all the criteria described above before it is considered an intelligent agent [63]. For example, in [64] Gilbert et al. from IBM describe agent intelligence as the "degree of reasoning and learned behavior: the agent's ability to accept the user's statement of goals and carry out the task delegated to it". In any case, the main benefit of agents that achieve these (additional) requirements is that they can act on their own, both upon recognized events in their environment or also decoupled from external stimuli.

Regarding the agents' intelligence, a distinction between *reactive* and *deliberative* agents has been proposed by both Bradshaw [63] and Nwana [60]. According to them, the main difference is found in the presence (i.e. for deliberative agents) or absence (i.e. for reactive agents) of a symbolic reasoning model in the agents.

For the smart home domain, agent based systems in general promise to be a suitable approach for smart home control. However, not all the incarnations of intelligent agents are suited in the smart home scenario or some offer more tailored functions for this kind of application, respectively. In particular, the choice of a specific agent type has implications on how well (or if at all) the requirements of the smart home system can be modeled as an agent system and implemented following the agent paradigm. It is thus necessary to further analyze agent concepts and evaluate them for their applicability and use in the smart home system. As intelligence is a central point in the smart home, focus is put on how this intelligent behavior of a smart home can be best realized following an agent based approach. In the smart home case, key requirements are that agents are capable of rational behavior with regard to the information available to them and that they are able to execute tasks autonomously (i.e. without obligatory user interaction). Additionally, we expect the agent system to be able to react both to internal and external events, for example, internally scheduled tasks as well as changes in the occupancy (cf. Section 2.2).

Given the above criteria, a rather simple *reactive* agent might seem sufficient for the smart home application because it is able to sense the environment, act upon these percepts and can be implemented more easily as it needs not keep any model about the world it is situated in. Moreover, it does not require user interaction and can basically be capable of intelligently selecting the actions to execute. However, some problems arise with the choice of a reactive agent. First, it would be somewhat

unclear where the difference and benefits compared to an object-oriented system are. In fact, the effort for following an agent paradigm might even not pay off in the expected way. Second and more severe, reactive agents still are not capable of initiating and executing actions themselves without having to wait for external events. An additional shortcoming is the lack of any methodology that supports the design and implementation of reactive agents. Most notably however, the demand for proactive behavior is the main argument against the use of purely reactive agents.

As an alternative to reactive agents, *deliberative* agents can be used. These combine features from reactive agents (mainly the capability to react to events) with the feature of proactive behavior. An overview of such an agent's internals are shown in Figure 3.3. Thereby, deliberative agents form some kind of hybrid solution that was partly developed to tackle the problem that "neither agents nor the world they inhabit are static" as stated by Wooldridge in his work [65]. In this paper, deliberative agents (also termed *deliberate*) agents are characterized to first of all have an explicitly represented internal model of the world they are situated in. This model contains explicitly modeled facts and represents the knowledge of an agent. Nwana [60] points out that "deliberative agents derive from the deliberative thinking paradigm: the agents possess an internal symbolic, reasoning model and they engage in planning and negotiation in order to achieve coordination with other agents". It is thus the case that deliberative agents are to be seen only in a multiagent system context and do not represent a sensible solution as stand-alone components. Furthermore, deliberative agents operate within a continuous cycle of the steps observation/perception, deliberation/planning and acting. Deciding on actions and reasoning thereby build on a kind of logical inference. Finally, it is mandatory for this type of agents that they exhibit a proactive behavior. Considering these characteristics, deliberative agents are in clear contrast to reactive agents as their design offers much more functionalities that can be exploited in an implementation. Nevertheless, also deliberative agents have their own shortcomings. Obviously, agents that can react to stimuli and that can also take action themselves are more complex to implement than those that are purely reactive. Apart from complexity, also the demand for a world model may pose a challenge. The smart home setting is a continuous, dynamic and non-deterministic environment. In order to obtain a world model, the environment with all its facets must be translated into a machine-processable representation, which is a complex task given the environment's nature and characteristics. Additionally, the agent performance is influenced by deliberation and the complexity of the world model. To ensure adequate response times, the agent must (internally) be designed in a way so

that it comes to good decisions within reasonable time. The temporal requirements are defined by the application and may range from milliseconds to minutes. Finally, a major question of deliberative agents is the balance of an agent's reactivity versus its proactiveness. On one hand, also a deliberative agent interfaces with its environment and hence may react to events that occur in the environment. In this case, the immediateness or directness of an agent's response is a characteristic parameter. On the other hand, a deliberative agent has a "desire" to pursue tasks that are not triggered by any event but tasks that the agent has an inherent interest in pursuing (i.e., the agent's proactiveness). In a practical implementation, the challenge lies in finding a trade-off between both aspects so that events in the environment still have an influence on the agent and at the same time internal desires of the agent are pursued continuously. In other words, it is necessary to find an acceptable trade-off between the fact how strong/long goals are pursued opposed to how directly/often the agent responds to external events. The question of finding an appropriate trade-off has long been an important topic in agent research, and has direct implications on how an agent is built, i.e. on the choice of an agent architecture.

Given the definitions and characteristics above, *deliberative* agents are identified as the most suited agent type for the application in a smart home control software. Main reason for this choice is the additional availability of proactiveness in this type of agent, which is a mandatory feature in the smart home when an agent is intended to take over control tasks from the users. Also concerning the energy efficient operation of the smart home, proactiveness is valuable as it allows to start energy-intensive tasks also without or prior to a direct necessity indicated by an event in the environment (e.g., a temperature dropping below a certain threshold). Furthermore, deliberation facilitates the specification of alternative actions that can be executed to achieve a particular goal. In the smart home, this characteristic can be exploited to model control (algorithm) alternatives for one smart home task, for example, specifying different cooling strategies for a room. Bratman [66] describes such an agent's behavior as practical reasoning. He states that practical reasoning is an approach which evaluates conflicting options under consideration of the agents desires and the agent's knowledge on its environment. Because this behavior fits best with the requirements of a smart home system as described in Section 2.2, agents in the smart home system will follow a practical reasoning approach for decision making.

The final question on agency is the selection of an appropriate agent architecture. In [68], Maes states that such an agent architecture is a particular methodology

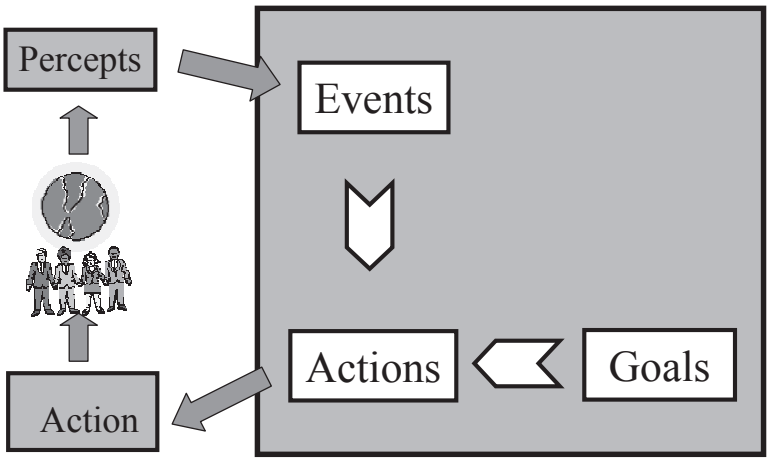

Figure 3.3: Agent diagram with reactive (*Percepts–Events*) and deliberative (*Goals*) components [67]

for building agents. It specifies how the theoretical concepts can be translated into a practical implementation of an agent. An agent architecture therefore implies a decomposition of the overall agent into a set of component modules and specifies how these modules interact and inter-operate to realize the agent's behavior. Taken together, modules and their interactions define which actions an agent takes upon consideration of its current internal and external states. In this dissertation, an agent architecture is understood and used as a translation of agent theory into agent practice in a computer system, a notion that is in accordance with Wooldridge and Jennings [69]. It is thus reasonable to follow an existing architecture as is provides basic building blocks that support the implementation of concrete agent systems building on a certain paradigm. Agent architectures exist for reactive and deliberative agents as well as for hybrid forms.

For our application scenario, deliberative agents following the BDI architecture (i.e. *BDI agents*) were identified to fulfill the requirements best. The reasons for this design decision as well as the basics of the BDI model will be explained in the following chapter.

3.2.2 BDI Agents

The Belief-Desire-Intention theory was developed by Michael Bratman [70] and at first conceived as a rather philosophical explanation of human practical reasoning.

BDI agents hence have their roots in the theory of practical reasoning. In his later work [66], Bratman explains the theoretical basics of practical reasoning for an agent as "a matter of weighing conflicting considerations for and against competing options, where the relevant considerations are provided by what the agent desires/ values/cares about and what the agent believes". Wooldridge [62] summarizes this explanation as reasoning that is geared towards finding the actions that are to be executed. Practical reasoning, as it is also done by human beings, features two distinct steps. First, one has to decide what we want to achieve, more formally, the future world state that we want to have. Once this goal is set, one has to think about and decide how to achieve this world state. In practice, this means to identify (a sequence of) actions that lead(s) towards this goal. Agents (in a software engineering meaning) that implement and follow such a way of deciding what to do are called *practical reasoning agents*.

At this point it is reasonable to differentiate practical and *theoretical* reasoning. Theoretical reasoning intends to generate new knowledge out of existing facts. It is similar to classic logics, such as description logics, where for example correctness of statements can be proven or implicitly describe facts can be made explicit. Theoretical reasoning is frequently used in the knowledge base of the smart home system as described in Section 2.3.2. There it is used, for example, to classify weather related parameters into previously defined classes of weather situations (e.g., airing-friendly weather). Practical reasoning is applied in the smart home control part, in particular within the agent system that is chosen to implement this system part. As stated above, this type of reasoning is used to find out which actions to execute in the smart home. Nevertheless, there will be a close interaction between the two system parts so that in the overall approach both types of reasoning will be exploited to achieve an optimized system operation. The smart home control part will for example draw knowledge for the knowledge base on the current weather situation to decide on which airing action to execute.

In order to implement practical reasoning agents, a computational framework is needed that supports, among other aspects, the reasoning process. One of the first frameworks that is based on the theoretical foundations that the works of Bratman established, was described by Rao and Georgeff in [71]. There, the authors took the theoretical BDI concept to a practical level. In this work, they also formulated several criteria and characteristics that – if met by a particular application

scenario – motivate the use of the BDI agent paradigm. The question that remains to be answered is in which cases one should opt for using BDI agents instead of any other agent type (e.g., purely reactive agents). To better justify the use of the BDI paradigm in the present scenario (i.e. a smart home control system), it is possible to compare this application domain with several criteria that were formulated by Rao and Georgeff for their dynamic and time-constraint reference system. As the authors state, the characteristics and limitations of their application are relevant for a large range of further systems of the same and related domains. Thus, if the application of an agent based smart home can be mapped to the criteria formulated by them, the feasibility of a BDI approach can be shown. Therefore, these reference criteria are cited from [71] and analyzed from a smart home system perspective in the following paragraphs:

- *[Rao and Georgeff]: "At any instance in time, there are potentially many different ways in which the environment can evolve (formally, the environment is nondeterministic)"*:
  In a smart home, changes in the environment may occur any time, are commonly not easily predictable and never completely deterministic. An example of the nondeterministic characteristics are indoor and outdoor environments which are by far too complex to be determinable by a smart home system. Similar, all events or actions that are in relation with human beings have to be considered nondeterministic. People enter, leave or act in a smart home without necessarily following a deterministic behavior pattern.

- *"At any instance in time, there are potentially many different actions or procedures the system can execute (formally, the system itself is nondeterministic)"*:
  A smart home control system has different possibilities to influence the environment as well as it has to manage multiple building services simultaneously. Thus, the system may choose among several different options to achieve a particular task. For example, actuators both for air-conditioning and for opening windows may exist. Among them, any one may be chosen to execute a cooling task. Likewise, the system potentially has the choice of taking actions in a variety of different services at any given point in time. For example, both a heating task and a demand side management operation may be available or desired for execution at the same time. In this case, the choice of the system is not predefined (i.e., deterministic), meaning that even if the input parameters remain constant, the selection of one option cannot be predicted with certainty.

- *"At any instance in time, there are potentially many different objectives that the system is asked to accomplish"*:
  The most prominent example of such a conflict of objectives in a smart home is found in the two main design goals of the smart home system: minimization of the energy consumption while providing comfortable environmental conditions to occupants. Obviously, not both of them can be realized *simultaneously* to full extent. For example, in some situations it may be beneficial to temporarily accept a slightly reduced user comfort (environmental conditions drop below a user comfort window), if in return a considerable amount of energy or a monetary benefit can be yielded.

- *The actions or procedures that (best) achieve the various objectives are dependent on the state of environment (context) and are independent of the internal state of the system*:
  When several options are available to achieve a smart home task, the system has to select one of them for execution. At any point in time, some different action may achieve best results or be applicable at all. The selection of an action by the system depends on (pre-)conditions of the actions and the state of the environment and in particular the system's knowledge on them, but on no further parameters/conditions. For example, the smart home control system can use either air-conditioning or ventilation to influence indoor temperature. The best suited action depends on the context of the action (e.g., the wind may be too strong to currently allow opening the windows) but not on the internal state of the computational system.

- *The environment can only be sensed locally (i.e., one sensing action is not sufficient for fully determining the state of the entire environment)*:
  In a smart home, this characteristic is fulfilled already because of the complexity of the environment the control system is set in. Single (and in practice neither multiple) sensing actions provide locally valid information, but never a complete model of the external world or some of its parameters. For example, a temperature sensor of the smart home provides only localized information on temperature in one room, but not a global indoor temperature. To obtain the temperature gradient of a whole room, at least several measuring actions have to be executed[2].

[2]In reality, the environment of a smart home (i.e., the physical world) is of overwhelming complexity. Depending on the granularity with which a globally valid model of a particular parameter

- *The rate at which computations and actions can be carried out is within reasonable bounds to the rate at which the environment evolves*:
  The system reaction time is the most difficult characteristic to provide in a smart home. Nevertheless, it is an important characteristic as it differentiates agent based systems from object-oriented ones. Main reasons for the challenge that is found in guaranteeing a reasonable reaction time are that control decisions can be computationally expensive and that the rate in which the smart home environment (i.e., the physical world) may change ranges from milliseconds (e.g., lighting) to minutes (e.g., heating). However, this criterion intends to ensure that the system can take action within reasonable time and that the system remains functional even if the environment changes during the action selection process or the execution of an action. In the smart home, many actions and procedures are based on control loops that are executed constantly during operation. Depending on their complexity, computations can be more or less complex and therefore time consuming, which might be of concern in some applications. This problem is alleviated by the control loops being designed to be robust against "failures" in the environment (control loops work towards minimizing this error) so that they can also cope with unexpected changes and do not become dysfunctional when they operate on an old world state. Main drawback of the latter is a loss in control performance and thus a resulting disadvantage regarding comfort and/or economic aspects.

While the first five criteria can quite easily be evaluated as achievable in a smart home, the latter demand of an acceptable computational performance requires further considerations. Essentially, the criterion again boils down to an agent's balance between its reactivity and its proactiveness. If the system sticks with the proactive execution of some function that it selected, it may happen that the environment changes during the execution of this function. In the worst case this may mean that the particular function becomes a no longer good or valid choice at some point in time during the execution because of external changes (e.g., because some precondition is now violated). Concerning reactivity, an agent system in a smart home faces a similar challenge. Given a high amount of reactivity, agents can very well adapt to a changing environment as the smart home. However, due to the environment changing frequently, a high reactivity can lead an agent to frequently change its evaluation of the "best" applicable function. Hence, it can happen that a function is executed

shall be constructed, it might thus even be the case that an infinite number of measurements would be needed for an accurate model. In practice, such a model can never be established.

one for a short period of time before it is superseded by another, currently better one. Thus, the agent is less committed to achieving a particular world state than it is focused on always selecting the best alternative among its pool of options. Given the long inertia of many services in a smart home, such a behavior would probably lead to only insignificant changes in the environment (e.g., turning on the furnace for 10 seconds will not significantly affect the room temperature) and as an ultimate consequence to a failure of sensible agent based control.

As hinted already in the previous Section 3.2, the challenge is to find a trade-off between an agent's reactivity and its proactiveness. This is a problem for which no common solution that allows to universally solve this issue is yet invented. However, approaches exist that allow to at least alleviate the problem in such a degree that agent based systems become fit for real world systems. As for example Kinny and Georgeff [72] argue, primarily practical reasoning agents are capable of solving this problem, as they allow to finely balance the degree of an agent's commitment to actions versus an agent's reactivity to events and in some way even enable to exploit the dependency between both characteristics. The most common incarnation of this class of agents are agents that follow the BDI paradigm. This paradigm provides both the logical foundations as well as concrete methods to cope with this latter challenge and the other constraints faced in a real-world system such as a smart home.

The BDI agent model is characterized by its comparably simple and straightforward approach to model cognitive capabilities. It was first presented as theoretical model by Bratman [70] where the three main concepts of *Beliefs*, *Desires* and *Intentions* were introduced as the mental attitudes of an agent. The classical BDI agent paradigm is composed of and around these three core parts, which are defined as follows [73].

- *Beliefs*: Beliefs reflect all information an agent has about its environment, on its internal state or also on other agents. A belief can either be some particular knowledge or a piece of information about the environment. In most cases, an agent will not carry information about all of its environment but rather only of those parts that are important for an execution of its tasks (e.g., a temperature control agent might not know anything about the light intensity in a room). This is not last also necessary in most real-world applications since a comprehensive view of all aspects of an agent's environment is almost always not possible. Thus, an agent may be modeled to only have a subjective

model of the current world which is an important point to notice when agent cooperation is targeted. Concerning the concrete implementation of beliefs, no limitations are specified by the paradigm so that any data structure can be used.

- *Desires*: An agent's desires stand for all world states that an agent wants to bring about. For this reason, desires are also called the motivational component of an agent. Desires thus represent objectives that shall be achieved in the system and in particular by the agent. An additional aspect of the concept of desires is that they enable a proactive behavior of agents because a BDI agent is conceived to pursue a selection of the set of desires on its own without any prior stimuli. It is important to note that an agent is likely to have many desires simultaneously and that these desires can also be contradictory at any point in time.

- *Intentions*: Finally, an agent may not be able to pursue all desires simultaneously, as at the bottom end physical actions in the environment have to be executed (e.g., a radiator valve must be opened to heat up the room). Therefore, an agent has to choose a particular set of actions (often called *plan*) that the agents believes will bring about (one of) the desired world state(s). This selected course of actions is called an *intention*. An intention is a desire that the agent currently has committed to achieve.

The resulting first model of a BDI agent is shown in Figure 3.4. It has capabilities for representing information (cf. Beliefs) and also for putting its desires into practice (cf. Plans). However, by strictly following only the initial definitions of the BDI paradigm as explained above, some problems arise or remain unsolved. First of all, an agent's desires may be contradictory (e.g., a smart home agent responsible for thermal comfort may have the desires to increase the temperature as well as to lower it, obviously depending on the current room temperature). In other words, the world states that an agent wants to bring about need not be consistent and could therefore imply contradictory agent actions. To overcome this problem, BDI agents implement a specific selection function on the set of desires. The associated process is termed *goal deliberation* and responsible for selecting an at the moment consistent subset out of all desires, then called an agent's *goals*. This set contains only possible world states that could potentially be reached simultaneously. Hence, goal deliberation is a central characteristic of BDI agents because it contributes to their artificial intelligence. Therefore, it is neither trivial to design nor to implement. In practice,

most BDI frameworks leave the implementation of this process to the developer who is responsible to develop a mechanism that matches the characteristics of the current application scenario.

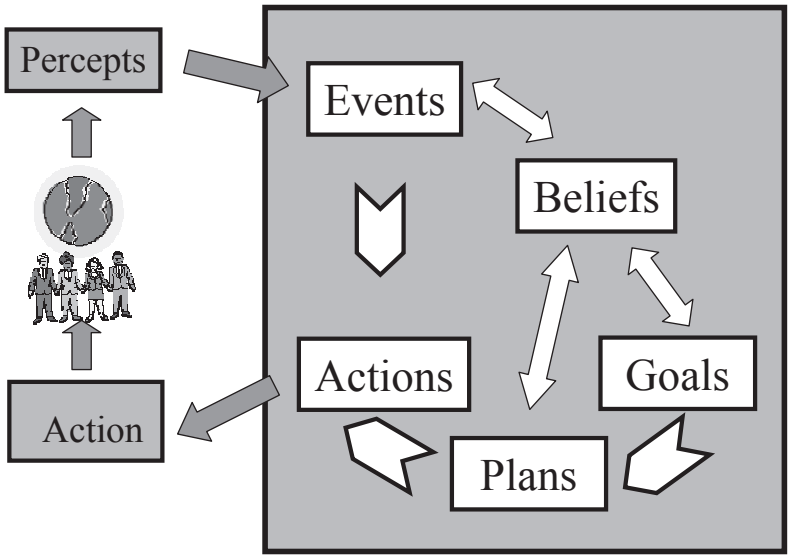

Figure 3.4: Agent diagram of an agent with *Beliefs* and *Plans* [67]

The second problem of the initial BDI definition is that it does not yet fully specify how goals are achieved. A set of actions that can be executed by the agent in the environment to reach one (particular) goal is called *plan*. Plans can hence be summarized as "means of achieving certain future world states" [74]. However, especially in a dynamic and complex environment many roads to success may be available, i.e., more than one plan may exist to reach one goal. Depending on the current state of the environment, the beliefs of the agent and maybe further knowledge, such as predictions on future world states or events, some solution (i.e., plan) may promise better (i.e., faster, more accurate, ...) results than another one. Thus, the challenge is to decide **how** the future world state (that is referred to as goal) should be reached. This decision process is called *means-end reasoning* (or sometimes meta-level reasoning). Within an agent, means-end reasoning is the process that is responsible for one of the plans becoming an intention of the agent. This selection process is in practice supported by plans carrying explicit pre-conditions that facilitate the means-end reasoning process. It thus incorporates the context in the decision process and ensures that a plan is currently applicable. For example,

in a smart home some cooling plan may be only executable when the outdoor temperature is below some threshold value. This requirement would be encoded in the plan as pre-condition.

Summarizing, agents that implement both goal deliberation and means-end reasoning capabilities are termed practical reasoning agents. Given these two additional features, an agent is able to decide which goal(s) to achieve, how to achieve these goals and is furthermore able to execute the basic elements of the intentions, the actions, in its environment.

BDI agents were initially chosen for their ability to balance reactivity and proactiveness. The main component behind this feature are an agent's intentions. When executing a plan, in particular the single action steps of this plan, the trade-off between reactivity and proactiveness can be realized with the help of the number of plan steps that need to be executed before a reconsideration of the plan happens. If the plan is reconsidered after each step, a *cautious* agent is implemented [72]. With an increasing minimum number of plan steps to be executed before a reconsideration may take place, *bolder* agents can be realized. The choice of how cautious or bold an agent should be largely depends on the dynamism of the environment. The problem is that agents may be situated in a dynamic environment which may change continuously, in particular also during the execution of the (atomic) actions of a plan or already during the means-end reasoning process. Such environment changes may result in some plans (or goals) no longer being worth pursuing. Bold agents stick with their plan for a longer time, although the plan may no longer be a reasonable choice. On one hand, in the worst case such agents may even work against their goals for some time. On the other hand, a bold agent is more likely to reach a goal more quickly since it remains committed to it. Cautious agents are ready to drop or exchange their plans quickly, but may overreact to small or time-limited changes in the environment, hence dropping a plan too soon. However, they are able to react quickly to changes and thus are more likely to always select the best available plan (i.e., to recognize when a selected plan is no longer worth pursuing and reconsider its possible alternatives). Thus, the rate of dynamism in the world is the main influence factor regarding this critical balance and influences how strong an agent should be committed to a particular plan.

Figure 3.5 shows the complete abstract architecture of the BDI agent. Events in the environment are perceived and used to update the agent's beliefs (1). At the same time, events may also directly trigger some actions (1). Based on the state of the environment (i.e., beliefs and percepts), an agent can identify which goals

to achieve (i.e., perform a goal deliberation; (2)). Once this is done, a plan to be executed is chosen (3). Finally, the plans are executed stepwise in the environment (4). Following the execution, new beliefs may be generated, new events may be perceived and also new goals may become available.

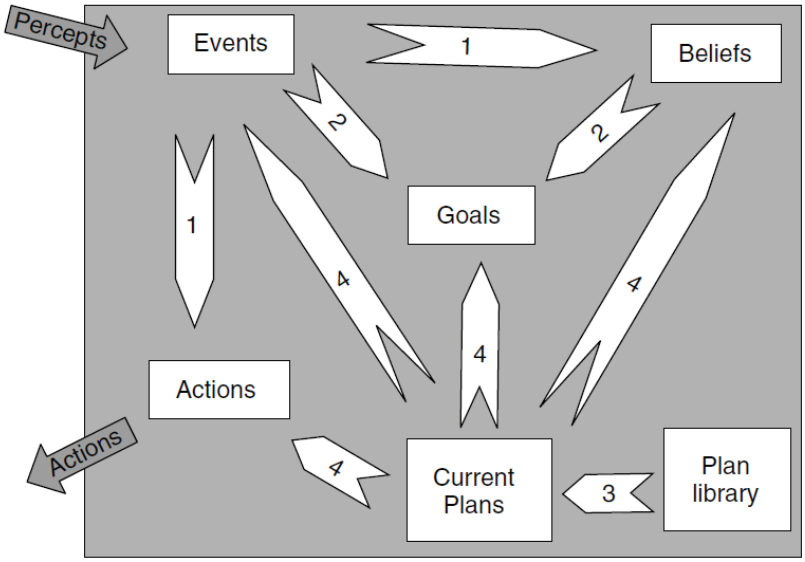

Figure 3.5: Agent execution cycle [74]

So far, BDI agents have been regarded and discussed as a system architecture but still independent from concrete implementation approaches. This concerns in particular the realization of the BDI core components beliefs, desires, intentions and the associated processes of goal deliberation and means-end reasoning. One of the first agent systems that implemented the BDI architecture was the Procedural Reasoning System (PRS) developed by Georgeff and Lansky [75], which was often extended and adapted, for example as a Java version for mobile agents in JAM [76]. Today, two main BDI agent platforms exist. JACK [77, 78] is a commercial agent platform frequently used in real-world projects, which offers extensive tool support for all kinds of tasks connected to the agent design and implementation process but requires the payment of license fees. Its main contender is the Jadex platform [79] which is based on the well-known (general) agent platform JADE. Jadex is available under an open source license and reuses as well as extends many of the advanced features already offered by JADE. For these reasons, Jadex is considered as the most suited candidate for an implementation of a smart home system.

Summarizing, BDI agents were identified as well-suited paradigm to realize a smart home system. Particularly the possibility to adapt the agent behavior to the smart home environment by exploiting the BDI features of goal deliberation and means-end reasoning complement the characteristics and demands of a smart home system well. In Chapter 4, a concrete agent system based on the BDI paradigm is designed and specified for an implementation in the Jadex platform.

3.3 State of the Art Agent-based Smart Homes

Agent based systems have been considered a scientific and also practical option as system architecture and implementation paradigm in various domains for the last decades. Regarding automation, agents were first used in the industrial automation branch, there controlling among others production processes and logistics (e.g., described by Lima in [80]) or enabling flexible manufacturing systems such as in the Pabadis'Promise project [81]. Due to their long time for evolvement, agents were and still are one of the software paradigms that are also practically used in real-world projects. Without any clear and significant reason apart from their increased complexity in the built environment due to the closer interaction with the users, this situation is considerably different in the "building-related" branch of automation. In the context of smart homes, building or home automation still only a handful of research projects (and none that are anywhere near commercial status) that follow the agent paradigm can be found today. While some projects use agents to conquer sub-problems (such as load scheduling or power management as outlined in [82]), a comprehensive agent based solution or at least an agent system architecture is not available yet. Nevertheless, some preceding projects exist that all have at least some relevance to the present agent based smart home system project.

Among them, the MavHome project which was described, for example, in [83] is quite well known. According to Cook et al. it is the goal to "create a home that acts as an intelligent agent" [84]. The authors propose a modular smart home architecture (cf. Figure 3.6) that on one hand focuses on the physical interaction with the help of a middleware concept. On the other hand, their architecture offers services for prediction, learning, policy enforcement and so on that are realized with the help of agent technology.

However, MavHome at the first glance looks like a specification for a multiagent based smart home, in a paper on the architecture of MavHome [85] the word "agent" is used only twice – in the abstract and the second sentence of the introduction.

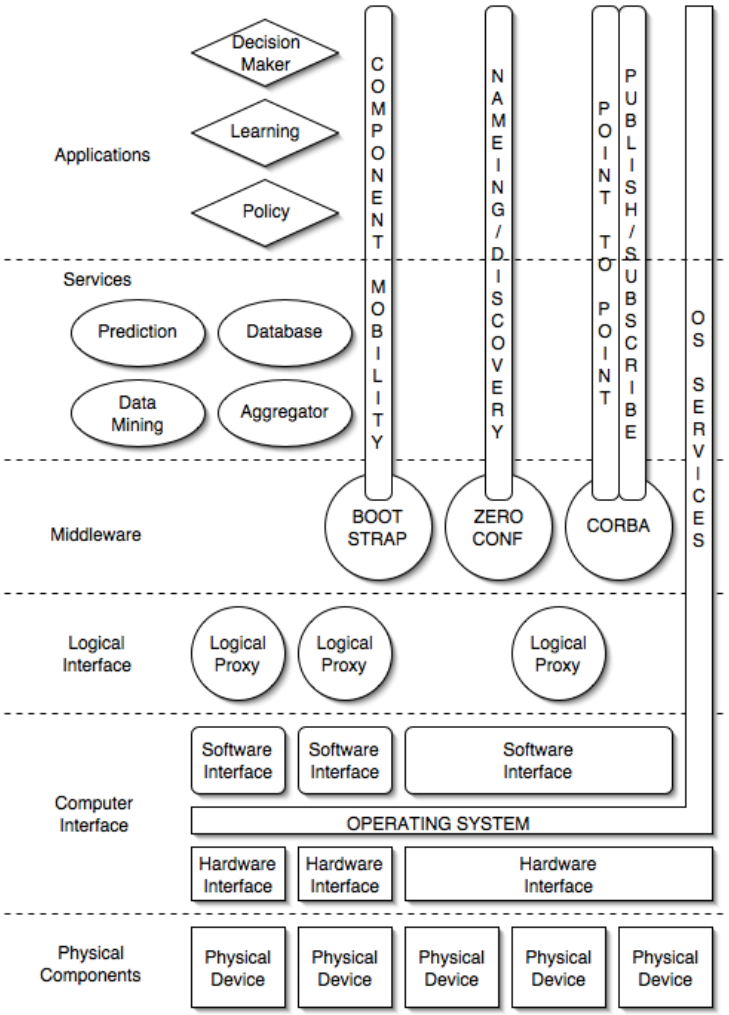

Figure 3.6: MavHome system architecture [85]

This illustrates that, although MavHome may be a valid approach of a smart home, it is not a thoroughly specified multiagent based smart home system. Rather, the whole MavHome architecture as shown in Figure 3.6 is conceived as one instance of a MavHome agent which implements all services (cf. Figure 3.7). Hence, it can be concluded that the authors have a different (compared to the definition given in this dissertation in Section 3.2) understanding of an agent and an agent system, as each of their agents implements the complete system architecture.

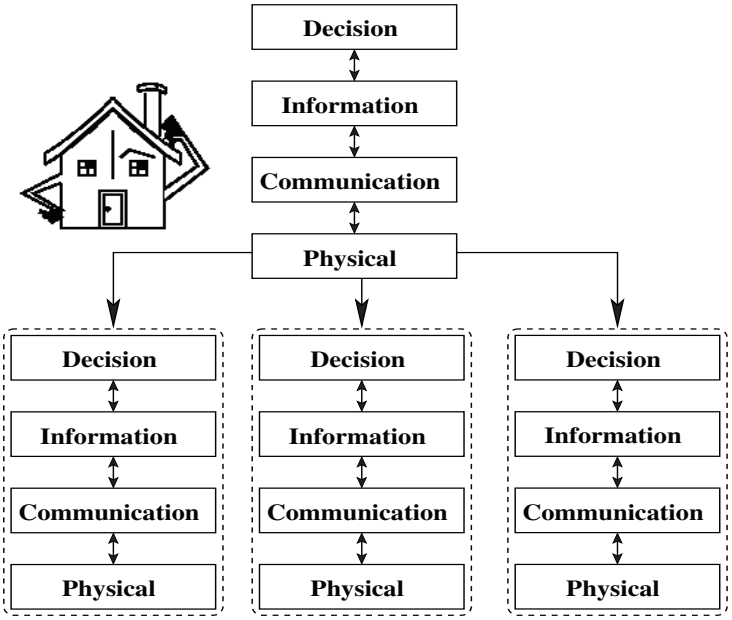

Figure 3.7: MavHome agent [84]

Another related project is MASBO, a multiagent system for building control [86]. This project focuses on the optimization of user comfort and energy efficiency in a building by applying agent technology. The goal is that one type of agents advocates the users' desires and preferences in the home, while other agents aim at translating these requirements into control strategies of the building automation devices. An overview of the MASBO abstract agent architecture is shown in Figure 3.8. It can be seen that the MAS approach adds enhanced functionalities, such as learning, decision making and reasoning to the building automation system. In another publication on MASBO, focus is put on agent negotiation to reach consensus of overlapping or contrasting agent actions [87].

In MASBO, agents are based on the specific EDA (Epistemic, Deontic and Axiological) model that works upon social attitudes and norms and includes further principles from semiotics[3]. Three different agent types are differentiated and abstractly described: *Personal agents* are representatives of the users in the multiagent system. They observe the environment, learn from and adapt to their associated occupant

[3] *Semiotics*: "a general philosophical theory of signs and symbols that deals especially with their function in both artificially constructed and natural languages and comprises syntactics, semantics, and pragmatics", Merriam Webster dictionary.

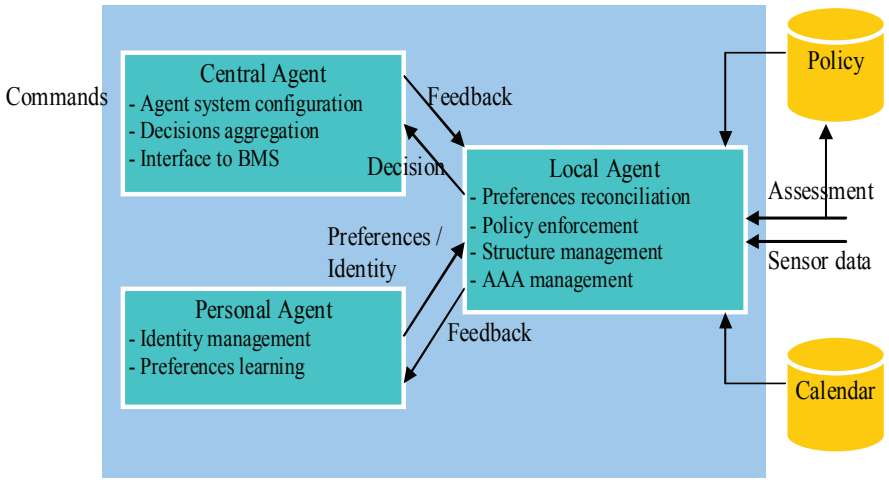

Figure 3.8: MASBO agent system architecture [86]

and provide feedback to the users. *Local agents* enforce policies, provide information, interface with the personal agents and resolve preference conflicts among policies and/or different users. *Central agents* are responsible for central system configuration, for collecting and aggregating decisions and they interface with the automation systems as well as further services.

The MASBO approach shares several aspects with the smart home system targeted in this dissertation, but cannot be considered a comprehensive agent system concept. Again, neither agents nor the agent system as such are described in detail but only on a very abstract level. Additionally, it is questionable if the uncommon EDA agent model that was first established in social psychology provides all characteristics (e.g., an agent platform to execute the agents) needed to successfully implement a smart home system.

The iDorm project [88] was developed by Hagras et al. at the University of Essex. iDorm is an intelligent dormitory that is used as a testbed for ubiquitous computing environment. In the iDorm approach, embedded agents are used to create an ambient-intelligence environment in the home. Focus is put on implementing system learning capabilities with the help of fuzzy logic controllers as well as on the development of a gateway server that abstracts underlying heterogeneous protocols. The learning architecture with its embedded agents is shown in Figure 3.9.

Similar to MavHome also iDorm understands agents in an embedded sense. Agents among other functions encapsulate learning functionalities, for example, a

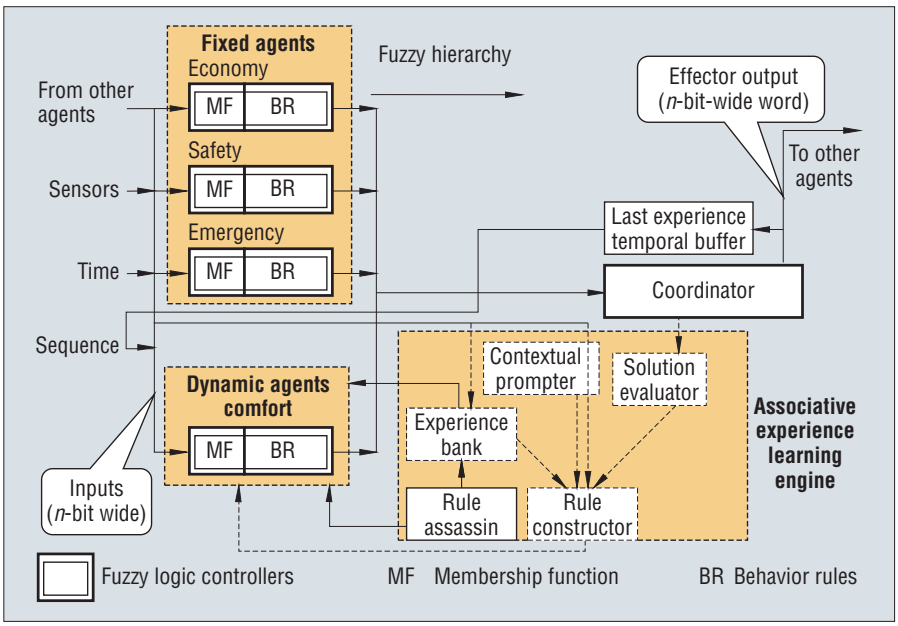

Figure 3.9: iDorm learning architecture with agents [88]

fuzzy approach of learning as presented in [89]. The overall system is therefore not necessarily only a multiagent system but agents only realize selected parts of the system.

Davidsson and Boman [90] propose a multiagent system concept featuring different agent types that control energy delivery and usage within an office building. Ultimate goal of their proposed system is a reduced energy consumption which shall be realized with the help of agents. For this purpose, four agent types are differentiated: *personal comfort agents* represent a particular person including his/her preferences and act in favor of this person during system operation. *Room agents* have the capability to control some environmental parameters of particular room and aim at an energy efficient operation of the equipment. *Environmental parameter agents* monitor environmental parameters of a particular room and can interact with sensors and actuators found in the room, thereby enforcing the values defined by the room agents. Finally, *badge system agents* track general occupancy and may maintain the location of particular persons in the building.

Figure 3.10 shows the published agent architecture of Davidsson and Boman. Their agents can thus act upon shared knowledge (*knowledge base*), have decision capabilities (*decision module*) and can execute actions that are organized in plans

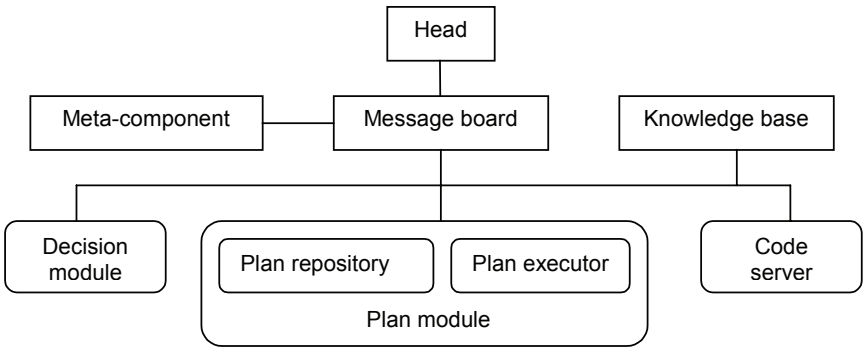

Figure 3.10: Agent architecture [90]

(*plan module*). Furthermore, a *head module* provides a communication interface, the *meta-component* manages addition or deletion of further modules, and communication among the agents is facilitated by the *message board component*. A particular characteristic of Davidsson and Boman's architecture is the separation of plans and their practical implementation as code. The latter is represented within the *code server*, from which the agents can fetch the (executable) code of the actions that compose a plan they want to execute. Additionally, agents have a set a preprogrammed rules that are encoded directly in the agents [91]. Davidsson and Boman implemented and tested their system system regarding the energy performance of a thermal comfort use case. Both a reactive and a proactive MAS implementation yielded significant energy savings compared to a classic thermostat based approach [92].

In [93], the authors describe an agent based smart home platform which focuses on context awareness and affective control. Main contribution of this work is a first concept of an agent architecture showing some agent types and their relations. The agent architecture of Benta et al. is shown in Figure 3.11. The agent system is composed of several main building blocks which each encapsulate one or more agents. The *reasoning and decision layer* can generate knowledge through inference strategies and is responsible for rule execution. The *context information layer* computes a context model and thus establishes context of actions and events for devices, objects and inhabitants. Finally, the *sensor* aggregates and analyzes information from the environment and offers this information to other agents, while the *action layer* executes selected actions in the environment.

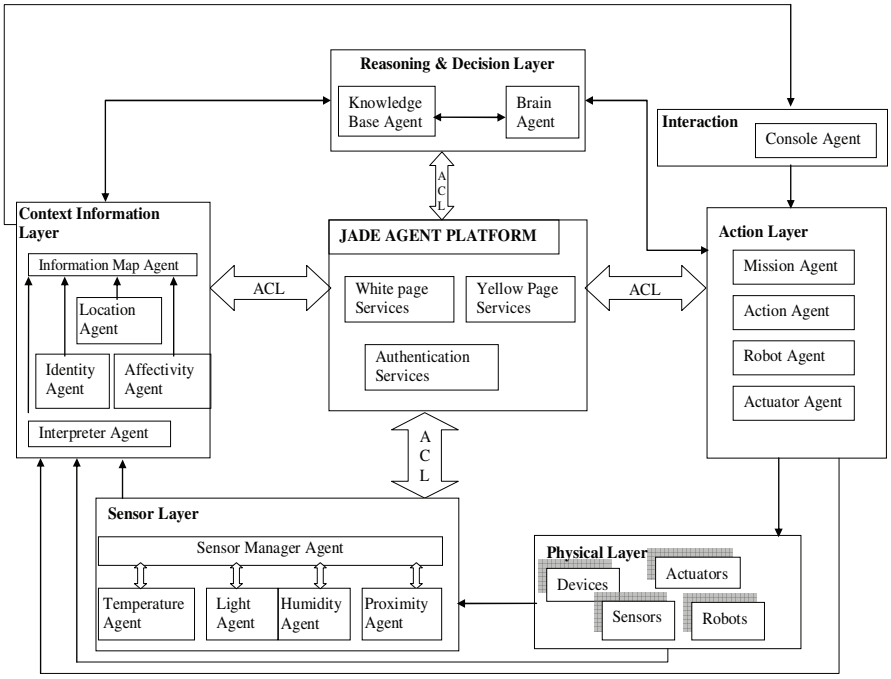

Figure 3.11: Agent architecture [93]

Furthermore, the system works on a knowledge base and specifies a user interface for system interaction. Although no details on the internal agent design are given, it can be assumed that the agent follow the BDI architecture since they are implemented using the Jadex framework for BDI agents. Furthermore, the agent communication language (ACL) which is part of the FIPA standard is used for message specification.

The multiagent home automation system (MAHAS) research project of Abras et al. [94] focuses on energy consumption optimization of appliances in buildings. It aims at exploiting load shifting and energy accumulation using an energy management system that is designed as and realized by a multiagent system. The system is modeled under the two aspects *service* and *satisfaction*, where services refer to the use of appliances and satisfaction provides a notion of user comfort. The multiagent system of MAHAS consists of one basic agent type of which an instantiation is embedded into a power resource or an equipment. The agents are internally designed as illustrated in Figure 3.12. With the help of prediction algorithms for power usage, weather and user behavior (usage) plans for the appliances are developed. Coopera-

tion among agents (*emergency mechanism*) is mainly used to increase user satisfaction and builds on a negotiation protocol [82]. The two different agent incarnations *equipment agents* (i.e., agents embedded in energy consumers) communicate with *energy resource agents* (i.e., agents embedded in energy producers) to optimize user comfort and energy efficiency. Similar to the project of Davidsson et al., MAHAS knows reactive and anticipative (i.e., proactive) control operation modes which differ mainly in their control horizon.

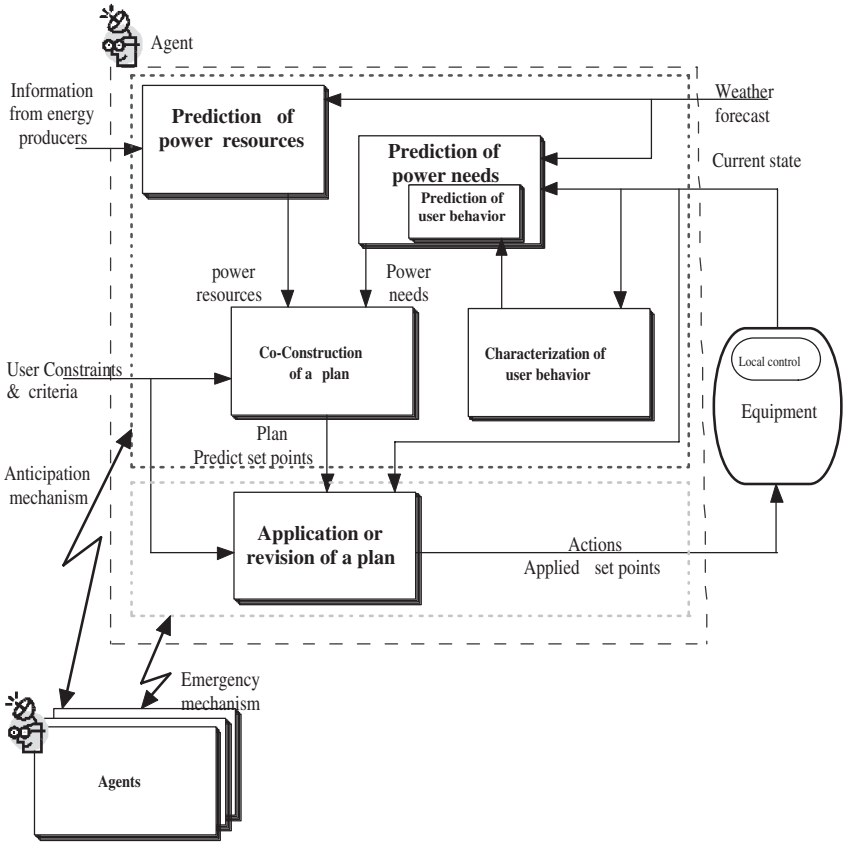

Figure 3.12: MAHAS agent architecture [82]

Mo and Mahdavi [95] propose an agent based control system that is tailored to resolve the conflict of user comfort and energy efficiency. For this purpose they introduce a "bi-lateral control scheme" in which global control is taken over by a building operator, but the individual users can fine-tune their settings to maximize their individual comfort. Agents are used to negotiate the control actions between users

and building operators, in particular to resolve potentially conflicting actions. The agent society consists of five different agent types, shown in Figure 3.13. The *operator agent* models the operator including his/her knowledge. Likewise, the *occupant agent* reflects its user in the system, can communicate with the human individual via a dedicated interface and may also implement learning capabilities. A utility agent provides connections to miscellaneous external systems, for example, a weather service. The *actor* and *sensor agents* are used to obtained information from and execute actions in the environment, respectively. The actor agent also implements functions to detect potential conflicts between the users' wishes and the operator decisions. In such a case, it starts a negotiation session. In this session, a negotiation protocol is executed which starts with an evaluation of the expected energy costs of the action. Furthermore, the individual discomfort of all possible action arrangements is calculated so that in the following deliberation phase an action that achieves greatest utility can be selected.

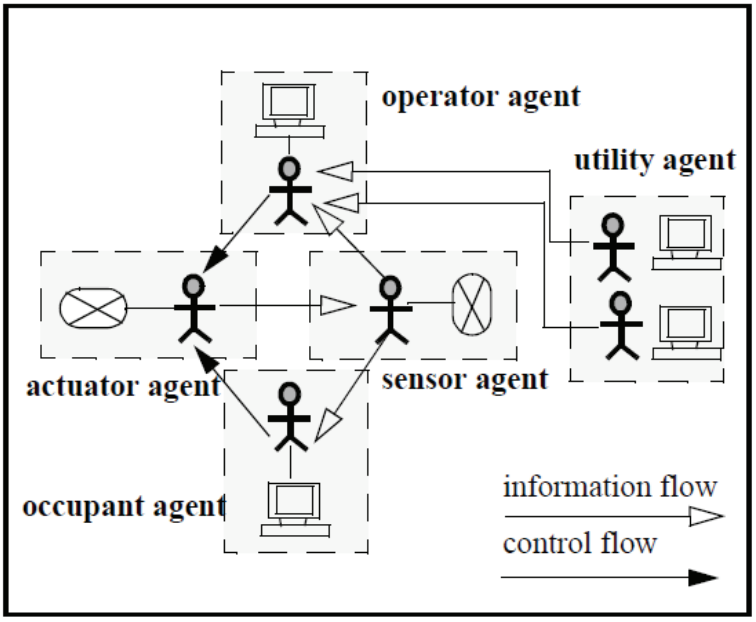

Figure 3.13: Agent types of the bi-lateral control system [95]

Finally, several smaller projects exist that at least claim the use of agent technology in the building environment. The *UT-AGENT* project [96] uses agents as avatars for the users. These agents advocate the preferences of their user in the smart home environment and thus are able to interact with a broad range of equip-

ment, e.g., TV, coffee maker or air-conditioning system. The agent design features profiles for each user and uses case-based reasoning capabilities to adapt to user preferences as well as to identify control actions. Another multiagent control system for energy efficient buildings has been proposed by Wang et al. [97]. Their multiagent approach is targeted to solving the conflict between resource efficiency and user comfort. Agents are used at two hierarchy levels: *central coordinator agents* interact with the power grid or other power resources while lower level *controller agents* represent user comfort goals in the system and interfaces with local device controllers. The central coordinator agent is responsible for establishing a trade-off between comfort and energy consumption and therefore implements a particle swarm optimization algorithm to compute the setpoints then executed by the local agents (cf. Figure 3.14). An agent based control architecture for smart homes based on an OSGi framework is proposed in the paper of Zhang et al. [98]. In a three layer architecture, home gateways on the lowest layer provide interconnection to various smart home devices and implement control strategies as *service agents*. These service agents observe the environment, reason on control processes and execute functions in the environment. Above them, regional management centers control the homes of their region by means of dispatching *control agents* to the underlying devices where these agents are integrated into the local OSGi framework as service agents. At the highest layer, gateway operators are able to design and create control agents that implement control strategies and can later be dispatched to the homes by the middle layer. Additionally, management agents are described that are responsible for the management of functionalities encapsulated in OSGi bundles.

The related work in agent based smart homes can be roughly classified into two groups. First, quite a lot of work exists that deals with the intelligence part of the smart home. Therefore, topics that originally came from the artificial intelligence (AI) research, such as machine learning, context awareness and AI based algorithms (e.g. for control issues) are in their focus. In most cases it can be observed that the application of existing, proven concepts to a new domain is dominant (e.g., generation of a fuzzy rule base for learning in [88]) over completely new concepts. The smaller second group of projects leaves out the intelligence issues, but uses agents primarily as a software engineering paradigm to build or often only enhance an automation system. Hence, agent types, the system architecture and all kinds of interactions represent the main topics. However, none of these previous works goes into detail regarding several highly relevant parts: the agents' internal design, the agent system architecture, the interactions among the agents including how and what kind of

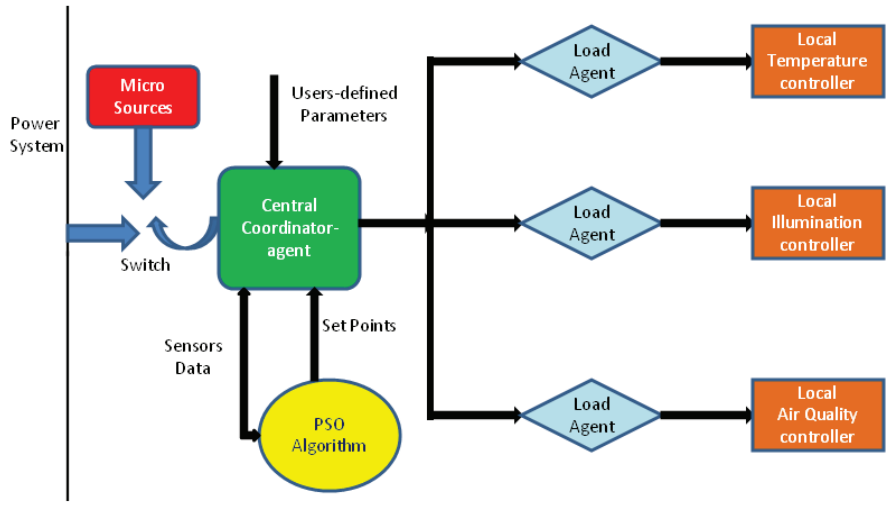

Figure 3.14: Wang's agent based control system [97]

information is exchanged as well as the overall system requirements and goals which should be the starting point for a system design.

Thus, one main point of critique of preceding research is that it fails to address the important topics in sufficient detail and that only fragmented system architectures/designs are available. None of the projects gives any details on the architecture of the agent system and the agents themselves. It is neither described what exact goals are pursued and how the plans to reach these goals are designed, nor is it clear if and how deliberation or reasoning processes take place. Often, agents are also specified in a generic way (e.g., sensor/actuator agents) missing any description of their functionalities within the multiagent system. Thus, agents are applied on a subsystems layer (e.g., I/O system) but they do not take over all system functionalities. Another important shortcoming is that in hardly any research project, a comprehensive approach towards system design is taken, probably due to its complexity. So, for example none of the projects states the use of any methodology, leading to an unstructured approach of system design and an often incomplete description of the system. One example is that agent types are informally (and in most cases also superficially) described and thus lack some of their most important details, such as a simple list of their goals. Furthermore, the communication among the agents of the multiagent system is only addressed in a selected few publications and at a high abstraction level at best. Hence, many issues and challenges of a smart home system, as previously raised in this dissertation, have not been considered in research

yet. This includes, for example, basic issues, such as a description how knowledge is represented in the system and how it can be accessed by the agents.

Summarizing, no satisfying *comprehensive* concept for a multiagent based smart home system can be found in related research today. Still, the projects that were analyzed in this section can provide valuable inputs for the agent based smart home system developed in this dissertation.

3.4 Methodologies for Multiagent System Design

For the design of a multiagent system several methodologies have been devised that support the design and development tasks. Methodologies can be regarded as (semi-) formal approach to design an agent system. They provide a structured approach and moreover guidelines which the designer can follow during the design process. This has the benefit of an eased design (or rather specification process) of the multiagent system as the design steps are already given by the methodology. Often also specific design artifacts, such as diagrams are a part of the methodology that facilitate the specification development. By following a methodology, the design progresses in the right (or at least a well-proven) order, so it is unlikely to leave out important steps or to miss fundamental design decisions, if not on purpose. Furthermore, the structured approach in many cases guarantees a higher consistency within the specification as normally less refactoring has to be done at later stages. Main points of critique of a methodology guided design approach are that by sticking too closely to a methodology, the view on important aspects may be obstructed, for example, performance issues. Obviously, also the methodology may not be flawless. In the latter case, even by closely following the methodology steps, the resulting design may have shortcomings.

Therefore, first of all it is essential to choose a methodology out of the numerous available ones that matches the criteria of the targeted application best. Second, it is often a more promising approach to not regard a methodology is the ultimate recipe that must be followed step by step without any deviation but to take a methodology as guidelines for a development process where single aspects or design artifacts may be more important than others for a particular scenario. In this section, several methodologies that have been proposed for the use in agent based systems are presented and analyzed for their applicability in the smart home system use case.

3.4.1 GAIA

The Gaia methodology was developed by Wooldridge et al. and described, for example, in [99]. It was later extended by Zambonelli with support from the original creators of the methodology Wooldridge and Jennings, which were his co-authors of [100]. Gaia was one of the first methodologies to be made publicly available and is still often cited in research works.

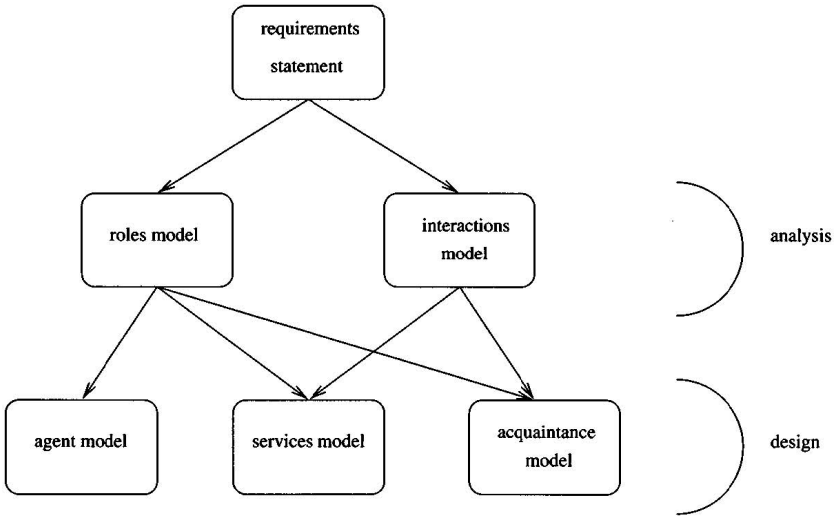

Figure 3.15: Gaia methodology stages and models [99]

Starting from a requirements description, Gaia differentiates two different stages: the *analysis* stage and the *design* stage. In the analysis stage the goal is to get an understanding of the system under design without going into any implementation details. Therefore, the system is analyzed for its roles and the interactions among these roles with the intention to gather further information on the specific tasks of the system. Accordingly, the two design artifacts produced by the methodology are called *roles model* and *interactions model*.

Each role in Gaia is characterized by *responsibilities*, *permissions*, *activities* and *protocols*. These capture all functionalities, associated permissions, tasks to be executed independently and interaction mechanisms with other roles belonging to one role, respectively. An example of a role schema in Gaia is shown in Figure 3.16. A specific detail of Gaia is the differentiation of the responsibilities in *liveness* and *safety* aspects. While liveness is used to model the activities during the regular system operation (i.e., "something good will eventually happen"), safety explicitly

refers to actions to be taken in case of unexpected conditions or other (system) failures (i.e., "nothing bad will ever happen"). In the interactions model, Gaia captures the dependencies and relationships among the different roles. Here the (interaction) protocols are defined in more detail so that all communication relationships in the agent system as well as all exchanged information are identified and specified. As during the analysis stage further details of the system may surface, Gaia advocates and iterative process which allows and even recommends several iterations of the analysis stage steps before moving on to the design phase.

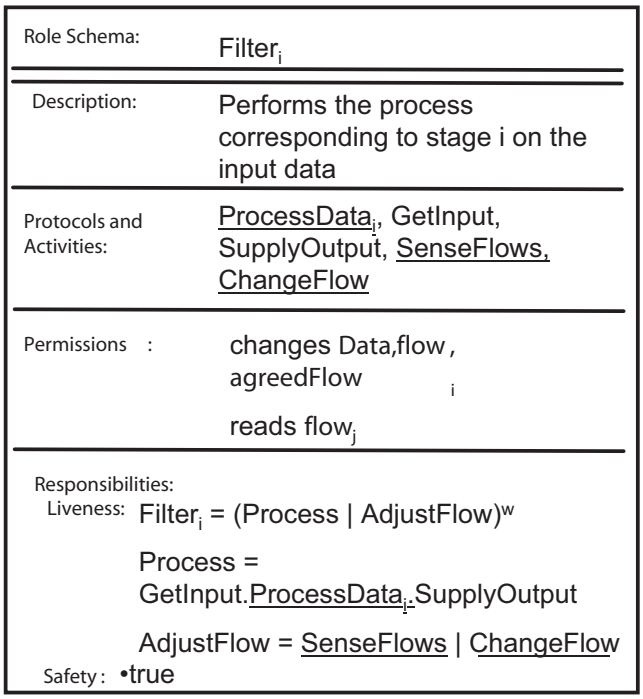

Figure 3.16: Gaia role schema [101]

In the design stage of Gaia focus is put on the creation of the architectural concept of the agent system as well as on the refinement and completion of the role and interactions models started in the previous phase. Starting from the abstract specifications, it is the goal to break them down to more low level concepts that are already related to a later implementation. The design process is split into three models (cf. Figure 3.15): the *agent* model, the *services* model and the *acquaintance* model. The agent model defines the different agent types needed in the system. It can be seen as a composition of the agent roles defined in the analysis stage. From this agent

model, a number of such agents (as software components) can be instantiated during system operation. In the services model, all services that will be available to the agents (i.e., that realize the agents' functionalities) are specified. These services relate to the activities identified in the analysis phase and also include specifics such as the safety requirements by means of pre- and post-conditions of services. Finally, the acquaintance model describes the communication and interactions among the agents in detail. It is represented by a graph that captures only the communication links but however does neither specify details on the exchanged data nor its semantics.

In the original version of the Gaia methodology [99], the methodology stops after the design phase and thus before tackling any implementation aspects. Rather, these were left open and the designer was referred to the use of established object-oriented techniques. In later extensions of Gaia (e.g., [101]), however, the actual implementation issues of the agents were at least partly incorporated into the methodology, for example, by providing guidance in how to derive class diagrams for the agents from the previous design artifacts.

3.4.2 MaSE (Multiagent Systems Engineering)

The Multiagent Systems Engineering (MaSE) methodology was described first in [102] by Wood et al in 2001. It is intended as a general purpose approach to design any kind of multiagent system explicitly including heterogeneous multiagent systems. MaSE covers the analysis, design, and development of such multiagent systems with a focus on agent types, agent interactions and also the internal agent design. For this purpose it draws from software engineering methodologies of the object-oriented ancestry of agents and furthermore relies on a number of different models that are accompanied by graphical representations.

Figure 3.17 shows the phases of the MaSE methodology. Similar to Gaia, also MaSE knows the two phases *analysis* and *design*. It is the task of the analysis phase to identify the roles in the system, where each role somehow contributes to achieve or at least get near the realization of a goal. The analysis phase again starts from the system requirements. First, goals are captured and organized in a hierarchical structure, leading to a goal hierarchy. Then, use cases are extracted also from the initial requirements and the communication paths are derived. These interactions are graphically represented in sequence diagrams which more or less automatically also identify the system roles. An example of the resulting role model diagram is given in Figure 3.18. As a final step of the analysis phase, concurrent tasks are identified.

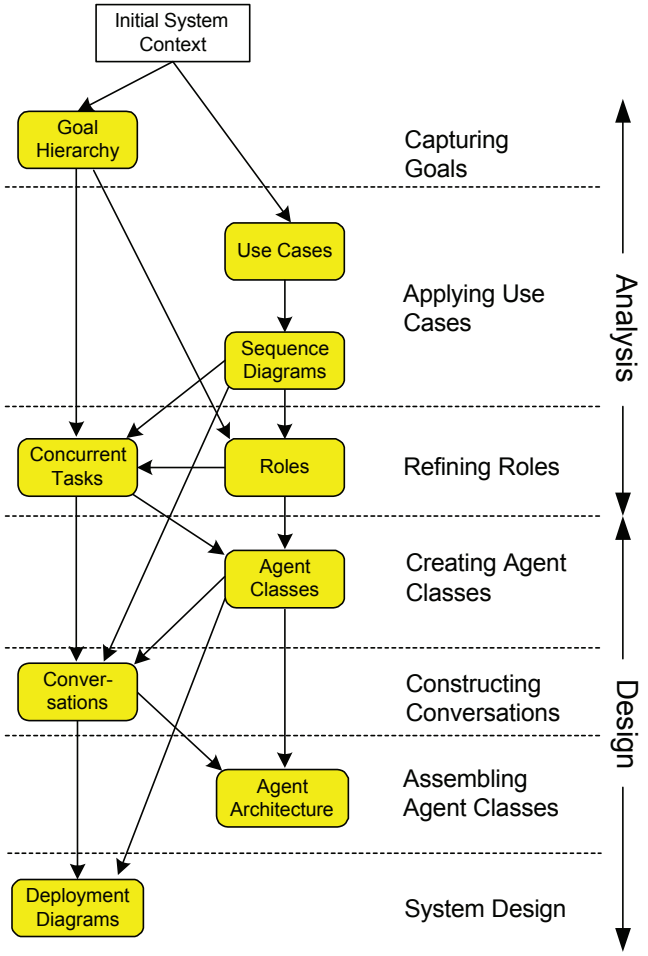

Figure 3.17: Phases of the MaSE methodology [103]

These describe how roles can achieve their goals and what kind of interactions with other roles are needed for it.

After the analysis is completed, the design phase starts. Here, first of all agent classes are created by grouping roles into *agent classes*, thereby aggregating functionalities. Additionally, also the relationships among the agent classes are identified and visualized in a corresponding diagram. In the subsequent step of constructing conversations, more details on the communication relationships among the agents are worked out. These are depicted in a communication diagram which formally is also a coordination protocol of the agent conversation differentiating between communication initiator and responder (cf. Figure 3.19). Unlike some other methodologies,

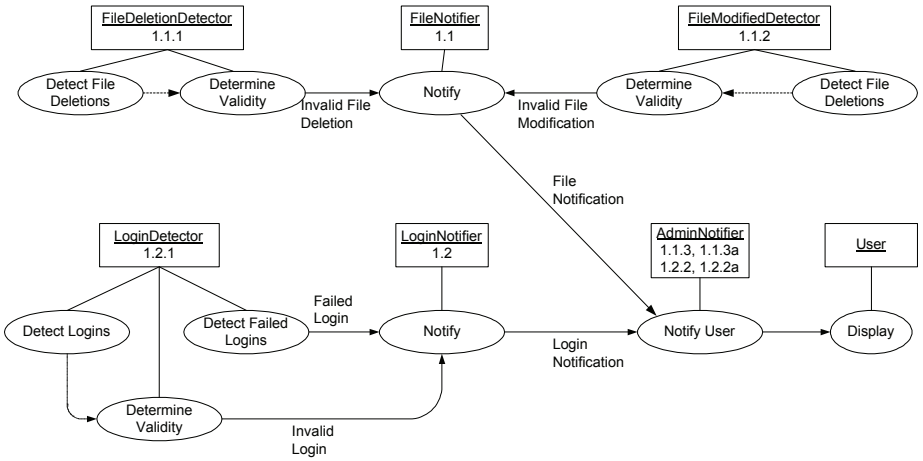

Figure 3.18: MaSE role model [103]

MaSE also covers the design of the agent internals. In the methodology, this step is called *agent architecture*. It describes the translation of the abstract specifications into a more concrete agent architecture, where however the choice of a particular agent architecture (e.g., BDI architecture) is left open to the designer. Thus, the architecture specification of MaSE remains quite abstract and resembles class diagrams known from object-oriented design. The final step of the MaSE methodology is the agent deployment which includes the instantiation of the actual agents. The associated *deployment diagram* illustrates among other aspects the cardinality of agent instances, the communication relationships broken down to the instances as well as the (potential) agent distribution to several physical systems.

3.4.3 Prometheus

The Prometheus methodology [104] was developed by Padgham et al. as a general purpose agent system methodology with a strong focus on its practical usability. It intends to be complete regarding the design and specification of an agent based system, i.e., it provides support for all steps towards a full system concept. Furthermore, the Prometheus methodology is one of the few approaches that is pervasively accompanied and supported by a software tool, the Prometheus Design Tool (PDT) [105]. One particular characteristic of Prometheus is its support for (automated) cross checking of the design artifacts. This is particularly necessary as Prometheus advocates to follow an iterative process of design and specification of an agent system.

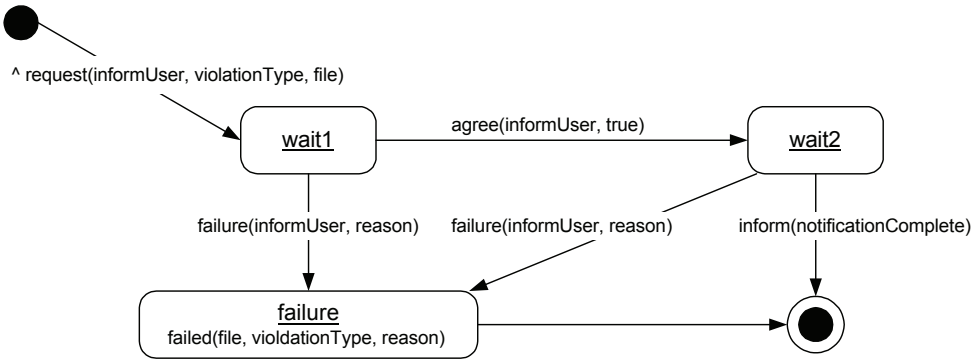

Figure 3.19: MaSE communication class diagram [103]

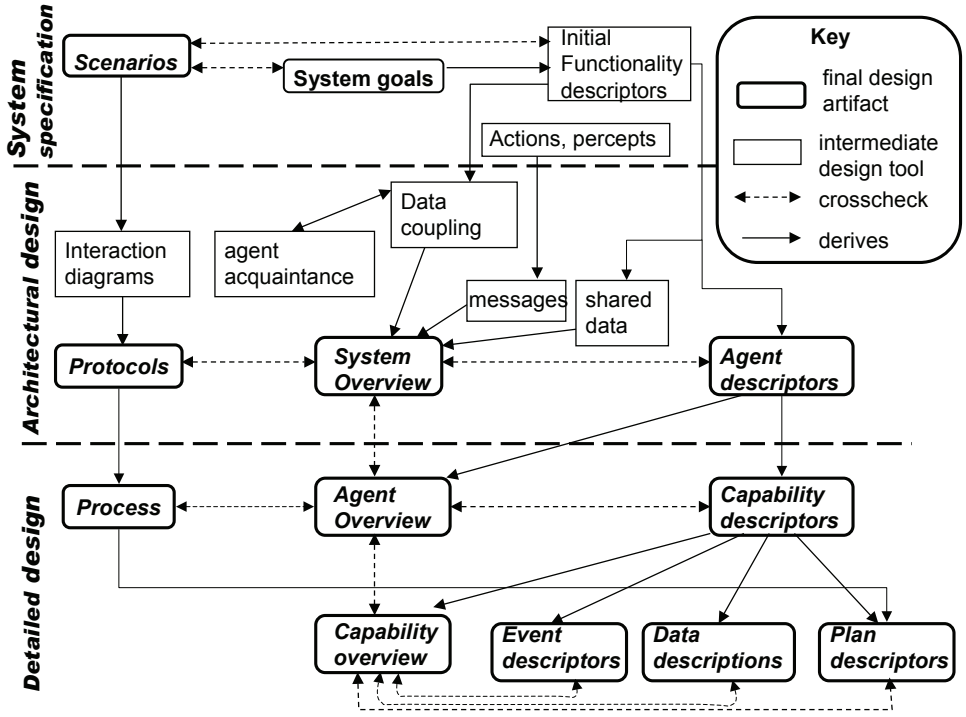

Figure 3.20: Overview of the Prometheus methodology [104]

As shown in Figure 3.20, Prometheus differentiates three phases of system design, the *system specification phase*, the *architectural design phase* and the *detailed design phase*. In the system specification phase, first the goals and associated use case scenarios of the system to be built are captured. Additionally, the interface of

the system to the environment is described by means of percepts and actions, and all previously mentioned building blocks are aggregated into so-called *functionalities*. Similar to other methodologies, in the architectural design phase again the transition from abstract system functionalities towards agent types is the core goal. Furthermore, the interaction relationships and required communication protocols among the agent types are specified. Two main design artifacts of this stage are the *data coupling diagram* and the *system overview diagram*, both are shown in Figures 3.21 and 3.22, respectively. While the first diagram provides a data centric grouping of system functionalities, the system overview diagram already illustrates a complete (yet still abstract) system view with all system components (agents, percepts, actions as well as their relations and their communications (messages, protocols, data resources) in a comprehensive way.

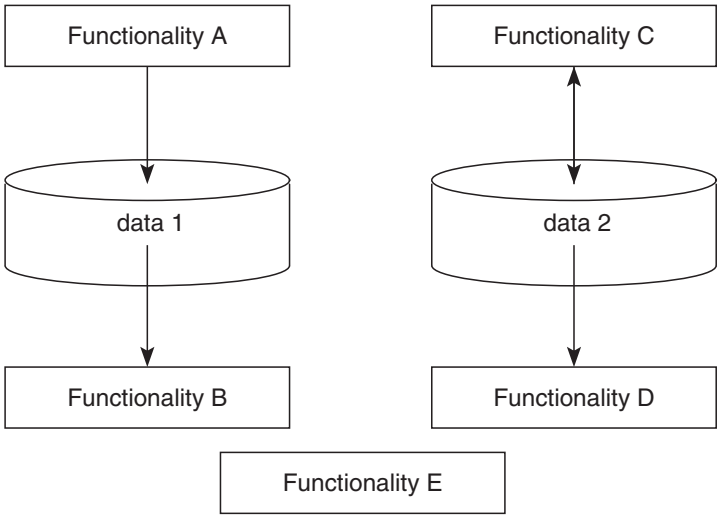

Figure 3.21: Prometheus data coupling diagram (abstract) [74]

The system concept is then refined and specified on a lower level in Prometheus' detailed design phase. In this stage, the internal design of each agent type is defined so that the system tasks can be fulfilled. Among other artifacts, the capabilities of an agent are defined by means of aggregating and merging the previously established functionalities. Important building blocks of capabilities are *plans*, a notion clearly related to BDI agents. Additionally, interaction protocols and plans are brought together by means of process diagrams, whose notation is closely related to activity diagrams.

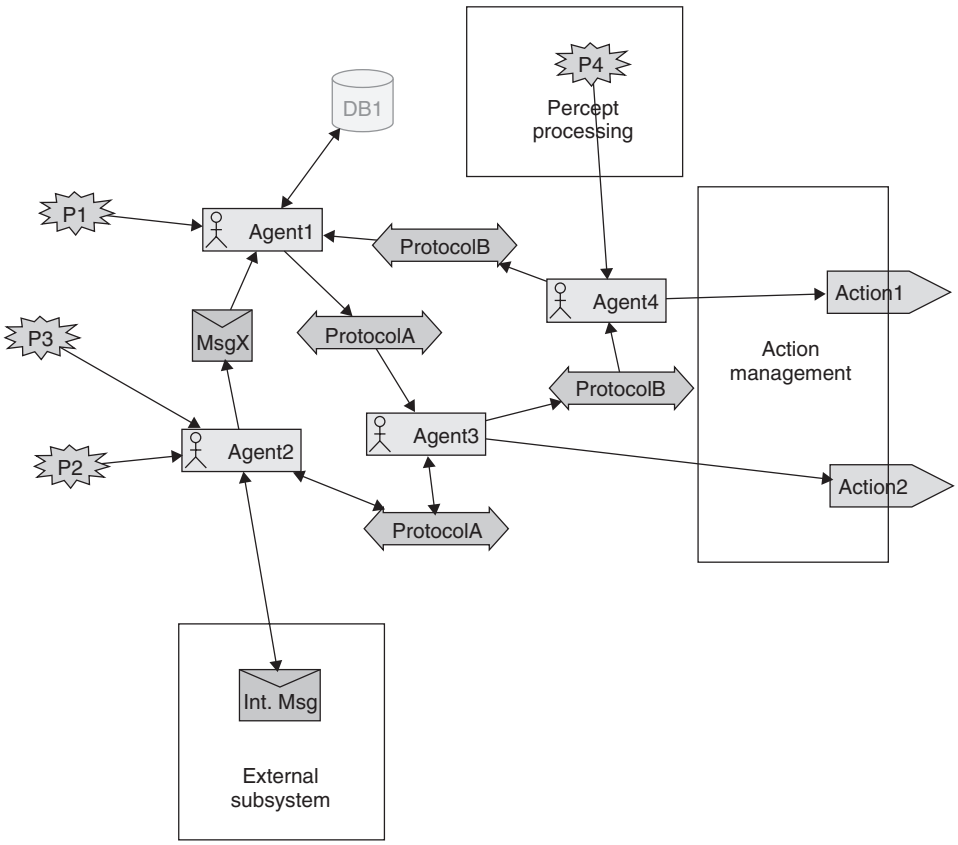

Figure 3.22: Prometheus system overview diagram (abstract) [74]

3.4.4 Tropos

The Tropos methodology was developed by Bresciani et al. in 2002 [106]. The creation of Tropos was guided by two main intentions. First, the methodology should cover also early design stages of a system. In particular, it should provide support already during the requirements engineering stage. This demand resulted in Tropos being not only goal- but also requirements-driven, i.e., it starts at the requirements analysis phase [107]. Second, Tropos (mainly) features agents with a "mentalistic notion", i.e., agents that have goals and plans. In the methodology, these mental concepts of an agent should be used pervasively during all design stages. Another characteristic of Tropos is that it heavily relies on the reuse and extension of concepts, diagrams and artifacts originally known from the Unified Modeling Language (UML).

Tropos differentiates between five development phase: *early requirements*, *late requirements*, *architectural design*, *detailed design* and *implementation*. During the early requirements phase, all classic requirements engineering steps are executed: requirements, system goals and stakeholders are identified as well as their interdependencies are captured. An example of a particular artifact of this phase is the goal decomposition diagram shown in Figure 3.23.

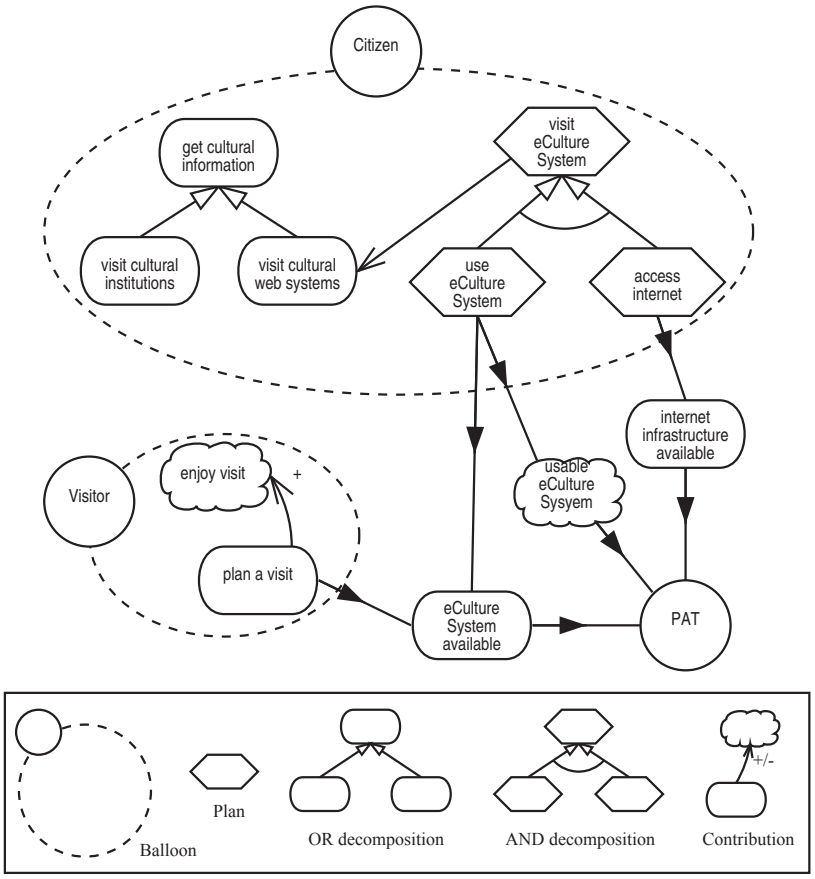

Figure 3.23: Tropos goal diagram [106]

In the late requirements phase, system and environment actors are introduced and specified in more detail. In other words the actors are set into context with their environment. In the following architectural design phase, the Tropos methodology knows the concepts of actors as sub-components of the overall system as well as it identifies the *dependencies* (i.e., data and control flows) among them. Additionally, the actors' *capabilities* that allow them to fulfill their goals are identified (cf. Figure

3.24). Based on the two previous steps of the architectural phase, the third and final step is to group actors and capabilities into agent types resulting in an abstract agent system architecture. In the detailed design phase, these agent types are specified in more detail and at a lower abstraction level. The Tropos methodology states that at this stage decisions on the agent architecture and development platform should have been made. However, Tropos features still quite generic artifacts that support this design phase. First, *capability diagrams* show a capability of an agent with the help of UML activity diagrams. Similar, *plan diagrams* illustrate the details of capabilities again by means of UML activity diagrams. Second, *agent interaction diagrams* specify communication acts among agents. These diagrams are based on the AUML (Agent UML) language, where agents correspond to objects. An example of this diagram is shown in Figure 3.25. Finally, the Tropos methodology is described in literature to also support implementation tasks, but publications fail to provide significant evidence to support this claim. However, practical examples are shown where the (BDI related) artifacts of previous methodology stages are mapped in a straightforward way to the (BDI) implementation framework JACK, in which methodological and implementation concepts even bear the same name. It can thus be expected that a transition from Tropos to JACK works, however it remains unclear if and to what extent other agent frameworks are supported.

Actor Name	**N**	**Capability**
Area Classifier	1	get area specification form
	2	classify area
	3	provide area information
	4	provide service description
Info Searcher	5	get area information
	6	find information source
	7	compose query
	8	query source

Figure 3.24: Excerpt from a Tropos capabilities list [106]

3.4.5 Comparison of Methodologies

The methodologies described above in some aspects follow similar approaches that might mainly be attributed to their common ancestry in design methodologies of common software systems. As such, in all agent methodologies a requirements en-

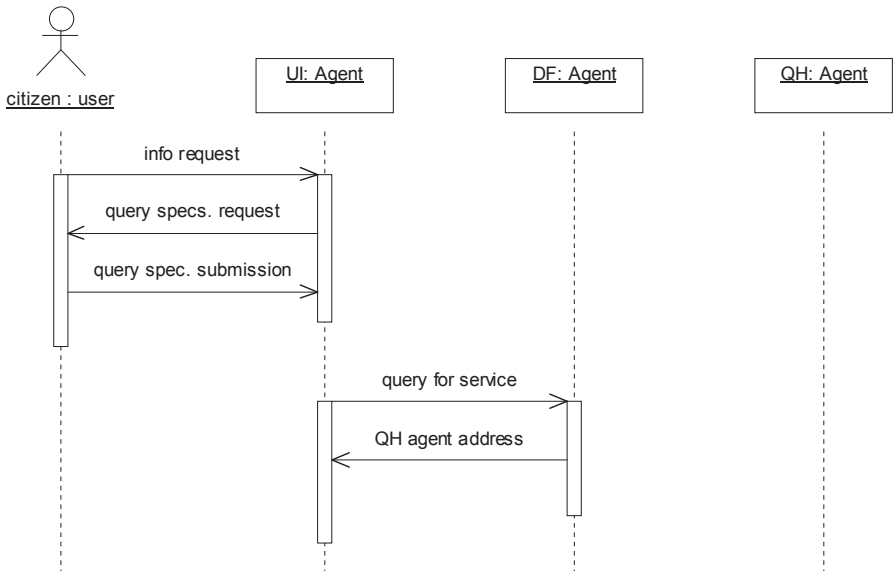

Figure 3.25: Excerpt from a Tropos agent interaction diagram [106]

gineering phase is planned at the beginning, which nevertheless differs in deepness. Most differences in the methodologies can be detected in the following agent design phase(s) which have the goal to first identify an agent system architecture and then to specify the agent internals. Depending on agent types and the agent architecture of the system under design, different approaches are taken as well as different characteristics are emphasized. Finally, implementation issues are often only rudimentarily covered, if described at all. In order to identify the most suited agent methodology for the design of a smart home system it is thus required to further analyze the methodologies regarding several criteria relevant to the smart home use case. An almost complete comparison (which however has not been made for the smart home case) of ten different methodologies for agent systems can be found in Chapter 10 of the agent-based methodologies book of Henderson and Giorgini [108].

First criterion is the *coverage* of the methodologies, i.e., how many parts of the engineering process are covered by it. A good overview of the coverage of different analysis and design stages of a system is shown in Figure 3.26. All discussed methodologies cover the core stages of late requirements analysis, where mostly a refinement of the requirements is targeted, and architectural design, where the agent system as such is created on a rather abstract level. It can be seen in the figure that in most cases the challenging process at the begin of the development of a new system, the

requirements analysis phase, is not covered by the methodologies (except Tropos). One reason for this fact is that from (traditional and object-oriented) software engineering many methods and proven approaches are already known and used for years. These can also be used when designing an agent based system since there need not be made any differences in, for example, the overall system description at such an early design stage. At the other end of the design cycle, the implementation related choices, such as the agent internals have to be decided. In this case, main modern methodologies offer at least some support for the developer. The challenge lies in the high heterogeneity of available agent architectures that may differ considerably in their approach. For this reason, methodologies that also cover the "detailed design phase" as termed by Giunchiglia most often make assumptions on the used agent architecture or remain very superficial in their specifications of an internal agent design.

Thus, another important aspect is whether the methodology is tailored towards a particular *agent implementation technology* and architecture or if it is a general purpose one. A typical representative of a general purpose methodology is MaSE, where neither assumptions on the agent architecture nor on the agent framework are made. Similar, Gaia does not really cover agent implementation in detail and hence can be considered general purpose, too. In contrast to them, Prometheus and Tropos focus on an implementation following the BDI architecture from the beginning and in fact tailor the final phases of their methodologies exclusively to this agent architecture.

Several standards for agent systems for example dealing with agent communication and interactions (e.g., FIPA standards) have been defined over the years. An up-to-date methodology should address and in the best case incorporate these standards into its design process to finally deliver standard compliant systems that can also be implemented more easily. Concerning the four methodologies discussed above, only Prometheus explicitly refers to the FIPA standards and bases its design on them.

Final aspect is the practicability of the methodologies in the real design process. This concerns on one hand the accessability of guidelines and the documentation of the methodologies and their details. On the other hand, pervasive support during their application is an important asset. While the first aspect is fulfilled by all approaches, the best support during the methodology's practical use is given again by Prometheus by means of the freely available Prometheus design tool that covers almost all stages of the methodology.

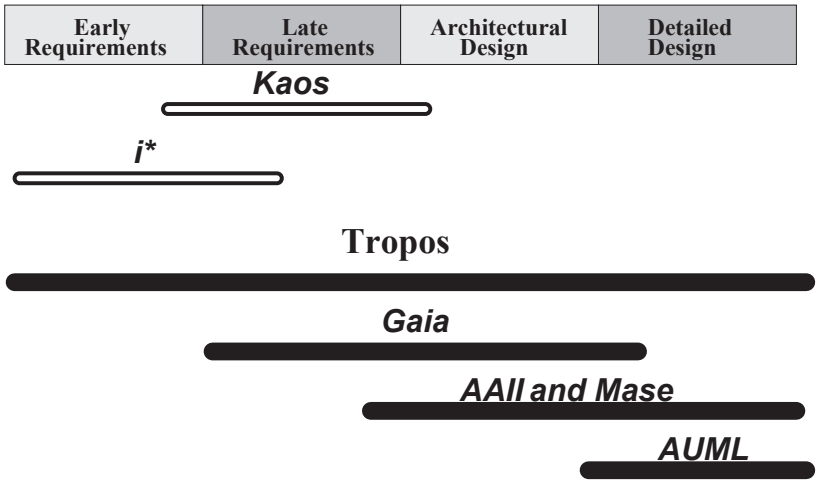

Figure 3.26: Comparison of Gaia, MaSE, Tropos methodologies [109]

Given the above characteristics of all methodologies, Prometheus is identified as the most suited methodology for the design (and implementation) of an agent based smart home system. This decision is also supported by the evaluation conducted in [108], where Prometheus performs well in almost all relevant criteria. It is a very complete methodology which focuses on the BDI architecture which was previously also identified as most suited for the smart home application. Furthermore, Prometheus provides excellent tool support, is tailored towards a practical implementation and supports consistency checking of the artifacts that are output during the design and specification process. However, although the completeness of Prometheus is generally a positive aspect, in practice the large number of detailed steps and artifacts can be a challenge in the design process. It is thus the task of the system designer to identify the most relevant steps for the given tasks and application (an approach which is also advocated by the creators of the methodology themselves, for example, in the text book [110]), in this way building a subset of the Prometheus methodology even more tailored to the current system under design. For the smart home use case, this will be described in Chapter 4.

Chapter 4

Prototype Implementation

4.1 Detailed MAS Specification

The detailed specification of the multiagent smart home system in this chapter follows in most parts the procedures defined by the Prometheus methodology. In a first step, Prometheus specifies the system specification phase where system goals and system functionalities shall be identified and their relations shall be illustrated in use case scenarios.

4.1.1 System Specification Phase

A first impression of the everyday operation of the targeted smart home system has already been given in Section 2.1. There, two user stories are found that sketch typical features of the system and show the system operation by means of example functionalities. These user stories closely correspond to the *scenario* artifact of Prometheus, however at a different level of formalization. Furthermore, detailed use cases have already been formally specified in Section 3.1 of this dissertation. These use cases represent all requirements of the system under design and provide well-suited input for both the scenario step as well as for the identification of system goals. In particular the latter design artifact of Prometheus can be derived in a straight-forward way from the use cases. This high level goal list which was created with the Prometheus Design Tool (PDT) [111] is shown in Figure 4.1. The goal overview provides an overview of the system's top level goals and features a first hierarchical grouping of functional (sub-)goals.

Main goal of the smart home system is the optimal smart home operation which is understood as a combination of the goals *energy efficiency, comfort maximization,*

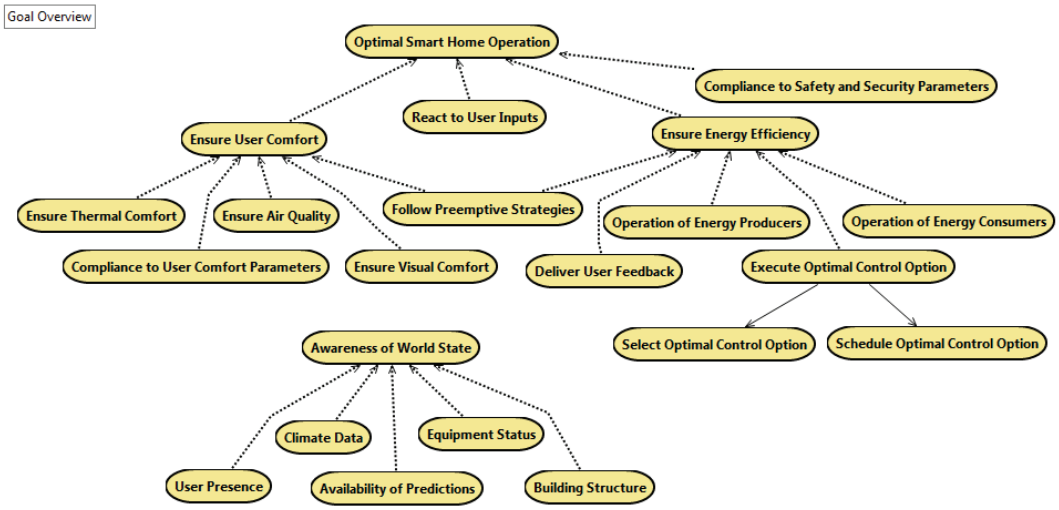

Figure 4.1: Functional goals overview diagram

manual control/override and *security/safety*. Comfort in the home is the sum of thermal comfort, visual comfort and good air quality, naturally taking into account the user's preferences. Energy efficiency is achieved when the user comfort is established with as little (energy) effort as possible. This includes the integration of (local) energy producers (e.g., renewable energy sources, such as photovoltaic installations), the selection of better/cheaper external energy sources, using the energy to execute the most efficient control strategies at the most efficient points in time and also the delivery of feedback to the user. Finally, the (agent) system can only achieve the other goals properly if it is aware of the environment it is set in. The goal of awareness is reached when current and future indoor and outdoor climate data is available, the (current and future) occupancy is known and when the system knows the built environment and the availability and status of all devices/equipment that can be used to influence the environment.

Next step in Prometheus is the identification of *system roles*. System roles group the previously specified single elements of percepts, actions and goals together into related pieces of system functionality. The roles hence model the relationship between the basic input and output as well as goals of the agent system. For the smart home system, the identified main system roles are depicted in Figures 4.2 and 4.3.

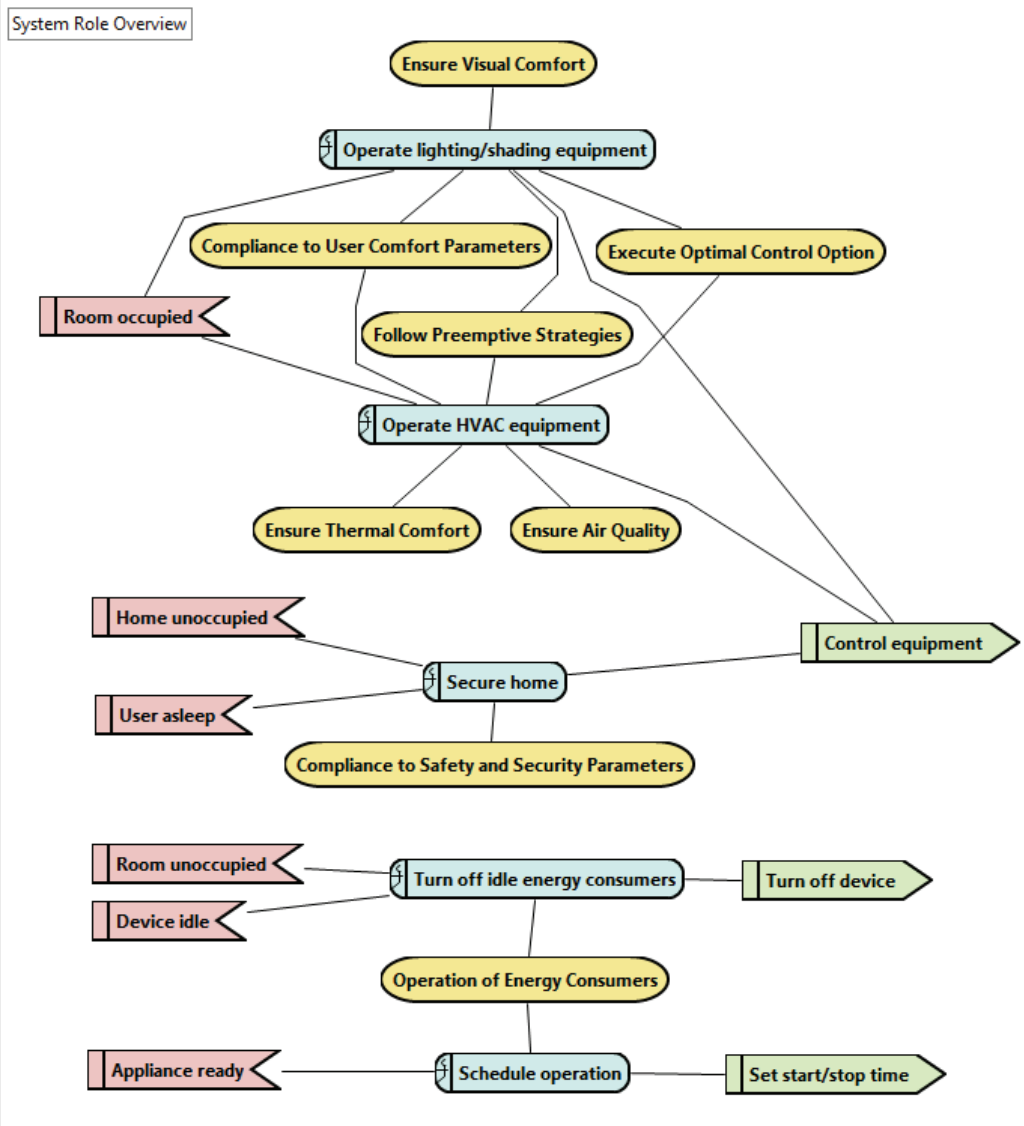

Figure 4.2: System roles diagram (upper part)

The system roles of the smart home system closely correspond again to the identified use cases. First of all, the system must be capable of operating HVAC (*operate HVAC equipment* role) and lighting/shading (*operate lighting/shading equipment* role) equipment as well as be able to control further (energy-consuming) devices (*turn off idle energy consumers* role) found in the smart home. Furthermore, system roles for the integration of locally produced energy (*local energy manager* role), the scheduling of device operations (*schedule operation* role), ensuring the security

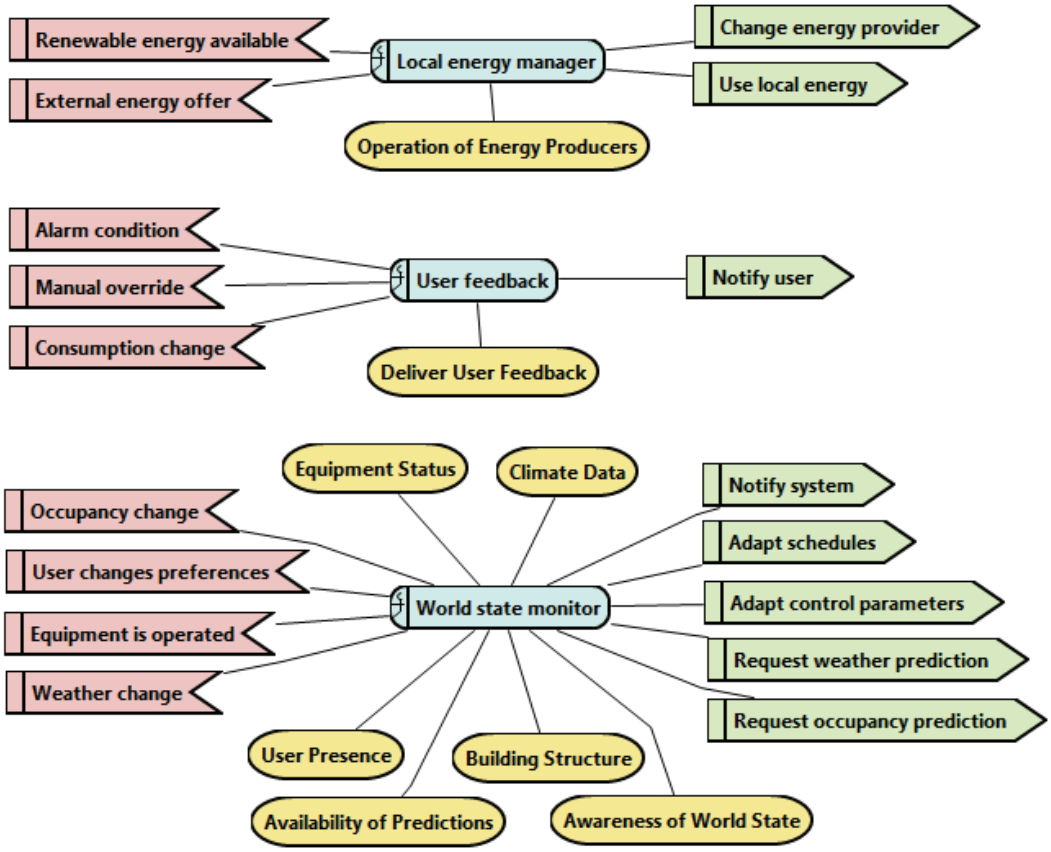

Figure 4.3: System roles diagram (lower part)

and safety of the home (*secure home* role) and for providing feedback to the users (*user feedback* role) are identified. Finally, in order to optimize the operation of the building services, the *world state monitor* role provides miscellaneous information on the environment the agents are situated in. The system role diagram also constitutes the final design artifact of the system specification phase of Prometheus. All basic information on system functionalities and system goals has been captured so that a precise understanding of the system under design exists. Starting from the developed artifacts, the high-level (i.e., not yet implementation related) architecture of the agent system can be defined. This is the task of the following architectural design phase of Prometheus and its associated artifacts.

4.1.2 Architectural Phase

The first important design artifact of the architectural phase is the *agent role grouping diagram*. It uses the different system roles that were identified in the previous step as input and groups these system roles together into agents. This results in a first definition of the *agent types* of the future system. These agent types provide an overview of the aggregated system functionalities that will later be implemented as one agent (instance). Although Prometheus does not demand a more detailed characterization of the different agent types, these are described in depth in the following paragraphs in order to provide a better understanding of the agent system.

- User agent
  Cardinality: 1 per smart home user
  Communicates with: Room agent; Global management agent; Data management agent
  The user agent is responsible for ensuring the comfort parameters of its owner within the smart home. It thus acts as an avatar of a human user in the smart home system. The design of the user agent follows the notion that to control the indoor conditions of a building in an energy efficient way, it is most important to reduce the control efforts to the lowest amount possible so that the users still feel comfortable. Therefore, this agent is aware of the preferences and the current and future presence of its user.

  For each smart home inhabitant, a dedicated user agent is available in the system. During the smart home system operation the user agent communicates and advocates the preferences of its (human) owner in the system. User agents may not only be created and stored in the system for permanent residents of the home but also for friends, personnel or other stakeholders that reside in the home more frequently. Since not all possible users of the smart home are known to the system in advance, persons that do not have a dedicated user agent (e.g., guests) are assigned an anonymous, temporary agent upon their arrival in the home. This generic user agent has standard comfort values which are predefined by the smart home owner and are not changed automatically during operation. User agents can also be used to limit capabilities of a specific user. For example, maintenance personnel may only be able to alter the comfort temperature within predefined boundaries.

  User agents for permanent residents are also permanently active in the system. The reason is that, e.g., for predictive control, it is required that the users'

preferences are ensured in the smart home control strategies although the user might not be at home yet. User agents for non-permanent inhabitants are dynamically instantiated when the user presence is observed by the smart home system or when their future arrival is actively communicated to the system, for example, by a human user. Such user agents are again deactivated when the guest leaves the smart home. Nevertheless, comfort preferences are preserved in the agent which is stored permanently in the system and is only removed upon manual intervention from the smart home owner(s).

The user agent not only acts on behalf of its human user but also monitors the associated person and communicates selected user actions to other agents. One example is the detection of a manual adjustment of the room temperature by the user. With the help of context awareness and learning algorithms the user agent can observe this action and interpret it. A learning algorithm can then be used to incorporate this observation into the stored comfort temperature profile, thus automatically adjusting to changed user preferences. This adaptation to the user is not limited to comfort temperature. Rather, the agent keeps profiles for all kinds of parameters and user desires, e.g., for multimedia preferences or air quality. The user agent furthermore keeps track of the habits of its user, for example, detecting and learning to never open a particular window or to turn on the TV on after return from work.

Finally, the user agent also provides (user-specific) feedback to the user if desired. This includes, for example, recommendations on how energy efficiency can be increased or feedback on the daily energy consumption. Similar, the user agent also provides the interface for the user to define or alter his/her preferences in the smart home system. As mentioned in the beginning, the user agent functions as a personal avatar of the user. As such, the agent is bound to a specific user but not to a specific location. In other words, it is possible that the user brings his/her user agent also to other smart homes that use a multiagent system. The user agent can be integrated there and perform the same functions (e.g., ensure user comfort) at any location.

- Room agent
  Cardinality: 1 per controlled room or thermal zone
  Communicates with: Room agent; Global management agent; Data management agent
  The room agent has the (control) authority over all building services and pro-

cess that are located in its associated room or zone. Hence, there is one agent for each room or zone that locally influences its environment. The room agent is the core point for the sustainable, energy-efficient operation of the smart home. It is responsible for the execution of the intelligent control strategies that influence the environment state. In particular, this concerns the control of thermal and visual conditions as well as the operation of miscellaneous equipment/devices in the smart home. For this reason, the agent is crucial for the realization of the global goals of energy efficiency and comfort. It aims at their permanent achievement taking into consideration user preferences, available building services and equipment and the current environment state. The latter point includes indoor and outdoor conditions, weather data, occupancy, locally produced energy and further parameters that can be used to improve control over the environment. For the computation of the actions that need to be ex-

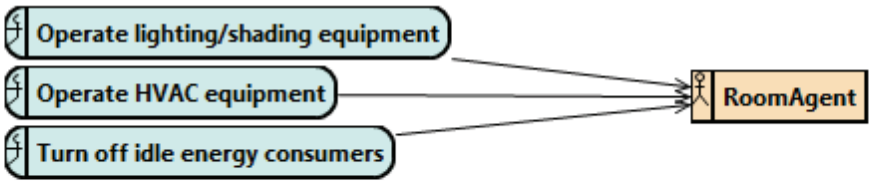

Figure 4.4: Agent role diagram – Room agent

ecuted in the environment, the room agent implements one or more different control strategies for its different control tasks. For example, the agent may implement a temperature control algorithm that uses "free cooling" and another one that uses the air-conditioning system. In a concrete situation, the selection of one specific control approach is based on the internal knowledge of the room agent and optionally also on further information that is requested from other agents (e.g., the data management agent). Similar, the room agent not only selects among different facilities to use in the control task, but may also implement different control strategies for the different equipment. Depending on the used control strategies, different information is considered in the computational process. For example, heating could follow a predictive control approach as well as be realized as simple on/off strategy. The advantage of this approach is that the room agent encapsulates the control strategies which can be altered, extended and also exchanged easily. This is for example necessary if new equipment becomes available in the smart home.

The room agent also controls the energy consuming facilities in the room. Based on the occupancy and the equipment type it aims at minimizing the energy consumption of these devices by activating energy saving modes or by separating the devices from the power grid. In order to provide fine-tuned control, the room agent manages a usage profile of the room, for example, to learn specific indoor environment conditions that are not related directly to the user, but more to the usage type of the room. These profiles also provide information on how to control the room in an unoccupied state, e.g., they are used to keep thermal conditions of a storage room within a certain range. This information is of particular importance for rooms that are typically unoccupied but can also be exploited for other rooms.

- Global management agent
  Cardinality: 1
  Communicates with: User agent; Room agent; Data management agent; Automation system agent
  The global management agent executes or manages all tasks in the smart home system for which a global view of the home is required. This concerns in particular safety and security, the integration of renewable energy sources and the handling of global energy efficiency strategies. It is furthermore the single point of access for external services that interface with the smart home, such as the smart grid, demand side management or performance contracting. Thanks to its global view, it is additionally capable of monitoring the overall energy consumption of the smart home as well as the consumption of selected devices or device groups. The information can also be visualized by the agent and be presented to the user. The agent also implements control strategies. In con-

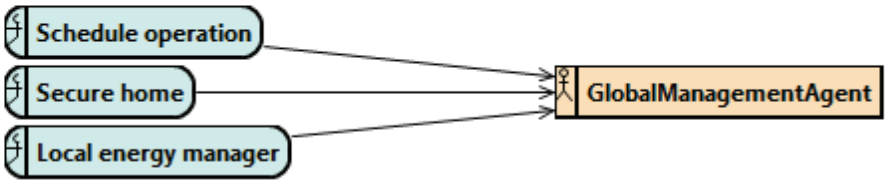

Figure 4.5: Agent role diagram – Global management agent

trast to the room agent, the global management agent deals with control tasks that are reasonably executed only in the whole smart home. One example is ensuring the security of the smart home when no inhabitants are present. The

agent not only has a global view of the persons that are currently present in the home, but it can also initiate actions in all smart home parts, for example securing all doors to the outside.

Another responsibility of the global management agent is the integration of locally produced energy. The agent knows all energy producing facilities of the home and can monitor their energy production. At the same time it is aware of all devices that consume energy and can thus ensure an efficient use of locally produced energy. In combination with the availability of deferrable loads (e.g., heat pump or dishwasher), the agent decides how and at which time such devices are operated. The global management agent can furthermore also request weather forecast data, predict the expected energy production of the day and compute an according operation schedule of specific devices. Similar to the user and room agents, also the global management agent encapsulates some artificial intelligence by way of keeping profiles on global conditions. Following the concept of global management, this concerns a (global) home profile that provides generally applicable conditions, e.g., a general setback temperature that may be overridden by the room agents. Other global information represented by profiles includes the general readiness of the users to conserve energy (i.e., how strong is the desire of the users to save energy, even although this means a trade-off with their comfort preferences) and information on security preferences (e.g., always lower blinds after sunset).

Finally, the agent regulates all external operations of the smart home in both directions, i.e., into the smart home and vice versa. This includes for example the distribution of locally produced excess energy when this energy cannot be used in the smart home for any reason but also negotiations with external energy providers concerning tariffs and energy amounts.

- Data management agent
  Cardinality: 1
  Communicates with: User agent; Room agent; Global management agent
  The main task of the data management agent is to provide information on the world in which the agents are situated (i.e., the smart home). The agent is aware of a variety of information on the world which taken together forms the world state. This world state or parts of it are communicated to the other agents upon request. The information that becomes available through the data management agent encompasses data on the physical environment as well as

information on the relationships among pieces of this data. Therefore, access to information on the available equipment, building, weather, energy providers and much more is offered by this agent. The information enables the other agents to execute their tasks (e.g., control strategies). Furthermore, the data management agent can be exploited by other agents to store data on behalf of them, for example to store and maintain the occupancy profile of the home.

Not all of these data is necessarily stored directly within the data agent. As outlined in Section 2.3, the proposed concept of the smart home system is designed to store these data in a dedicated knowledge base which is realized as a separate module. In this case, the data management agent can be regarded as a dedicated interface to all the information stored in the knowledge base. Details on the different stored data is given in Section 2.3.2.

The benefit of this approach is that a knowledge base provides mechanisms for data storage and also inference on these data. Compared to the rather simple internal structure of a BDI agent, especially its belief base, the knowledge base provides superior service in this respect. Furthermore, the decoupling of information representation and control components allows their independent evolvement. Still, the access to the stored data by the other agents in the agent system remains transparently possible without requiring any changes to the interface. The data management agent abstracts the external data representation and provides a generic interface to the other agents. In any case the agent retrieves requested information from its internal representation or the external data representation and distributes this information with the help of agent communication mechanisms.

- Automation system agent
  Cardinality: 1
  Communicates with: Room agent; Global management agent; Data management agent
  The automation systems agent handles all communication with the automation devices that are installed in the smart home. For the other agents it provides a generic interface to access all automation system functionality. It thus abstracts the underlying heterogeneous automation systems and relieves the other agents of implementing automation system specific functionality. It accepts standard agent communication messages and translates them into the communication types of the automation systems. The automation system agent

is also exploited to collect data from the sensors that are installed in the smart home and generally for obtaining all information that is requested by the agent system from any automation device. Thus, all communication to and from the dedicated automation systems passes through this agent.

So far the agent functionalities in the overall system have been described. However, in a multiagent system, communication among the agents is of great importance for the overall system performance. Single agents encapsulate related parts of the system functionality but mainly the system's intelligence is only realized through the interactions among the agents. The Prometheus methodology captures this agent communication in *sequence diagrams*. These diagrams are originally known from the Unified Modeling Language (UML) [112]. However, out of historic reasons, the UML variant called Agent UML (AUML) [113] which was specifically created for multiagent systems was and still is used in Prometheus[1]. The sequence diagrams of Prometheus capture interaction sequences among agents and are used to describe the so-called *protocols* between agents. The diagrams show a timed sequence of message exchanges and hence are very similar to their UML counterparts. Details on the notation used in AUML can be found in [114].

For the smart home system, the five most representative protocols are illustrates as AUML sequence diagrams in Figures 4.6 to 4.10. The diagrams have been created with the Prometheus Design Tool which provides a textual input method for these diagrams. Details on the textual notation of AUML that is also implemented in the PDT can be found in [115]. The PDT is capable of translating these textual inputs into AUML sequence diagrams. It furthermore supports consistency checking between the Prometheus design artifacts (e.g., the system role overview) and the AUML code. This prevents, for example, that agents that have not been modeled in Prometheus before, are used in the AUML diagram.

The first sequence diagram (cf. Figure 4.6) illustrates the communication that takes place when a user enters a room of the smart home. The user agent informs the room agent of the corresponding room that its owner, the inhabitant, has entered the room. The room agent now has the task to establish comfort conditions for this user. For this purpose it requests the user's preferences from the user agent. Additionally, it obtains the current room climate data from the data management

[1]The Prometheus methodology was created when UML in its initial version only provided sequence diagrams that featured a single interaction. However, for multiagent systems, often many interactions need to be modeled and represented simultaneously. Therefore, AUML was chosen for the use in Prometheus. Today, AUML is in many respects similar to UML 2.0 and no recent developments for AUML can be found.

agent. As indicated by the *opt* box, the latter two operations are only executed if the room agent does not already possess current climate and preference values (i.e., such a communication already occurred recently). In the following communication

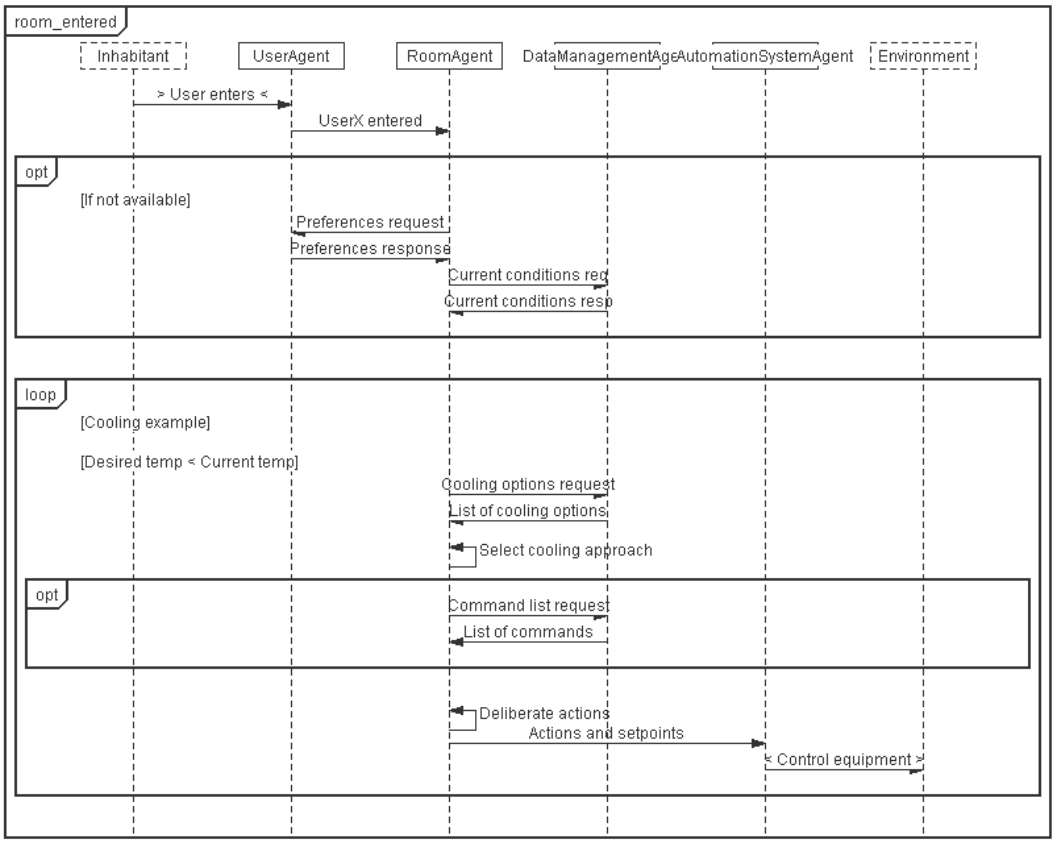

Figure 4.6: Sequence diagram – Room enter protocol

sequences, the example communication for a cooling scenario is depicted. As shown in the goal overview diagram (cf. Figure 4.1), user comfort actually includes thermal and visual conditions as well as air quality. For clarity reasons, these similar communication actions are omitted in the diagram. When a cooling task is detected by the room agent, it can request a list of cooling options for this room from the data management agent. Then is selects one of these cooling approaches that is currently applicable. This selection process is highly implementation dependent. It may follow user preferences (e.g., preferred devices), energy efficiency considerations, random schemes or even a combination of all of these aspects. The next optional step is the request of further details (e.g., executable commands) on the selected cooling

mechanism. This information can be used by the room agent to decide which actions shall be executed to achieve the goal of cooling down the room. An example is cooling with the help of an air-conditioning system. Here it might be the case that the air-conditioning system offers different cooling algorithms, for example, one algorithm that follows a PID approach and another one that offers only on/off functionality. In this case the room agent receives both options and can decide for one of them. Finally, the room agent sends its decisions and possibly new setpoint values to the automation system agent. This agent takes care of the execution of the commands in the underlying automation systems and thus the environment. The AUML code for the "room entered" protocol is given in Listing 4.1.

```
start room_entered
actor A Inhabitant
agent B UserAgent
agent C RoomAgent
agent D DataManagementAgent
agent E AutomationSystemAgent
actor F Environment
    percept A B User enters
    message B C UserX entered
box opt
guard [if not available]
    message C B Preferences request
    message B C Preferences response
    message C D Current conditions req
    message D C Current conditions resp
end opt
box loop
guard [Cooling example]
guard [Desired temp < Current temp]
    message C D Cooling options request
    message D C List of cooling options
    message C C Select cooling approach
        box opt
            message C D Command list request
            message D C List of commands
        end opt
    message C C Deliberate actions
    message C E Actions and setpoints
    action E F Control equipment
end loop
```

```
finish
```

Listing 4.1: Agent UML code for the "room entered" protocol

In Figure 4.7, the protocol that is executed when a user leaves the room is illustrated. The protocol starts with the user agent informing the room agent that the user left the room. The room agent internally has an overview of all people that are in the room. In the depicted case, nobody is in the room after the user leaves. Therefore, the room agent knows that it no longer has to guarantee specific comfort values but that it should now save energy. Hence, it requests the current value of the setback temperature from the data management agent and sends according commands for the HVAC system to the automation system agent. Since the room is not occupied, also different energy consuming devices in the room may be switched off. Although the room agent is aware of all devices in the controlled room, it requests power policy information from the data management agent for all these devices. This step is necessary to ensure that only unused devices are switched off but not those that require permanent power supply (e.g., the refrigerator). In a next step, the room agent also internally schedules the re-activation of devices. This allows the agent to temporarily switch off devices that only need to be active during certain times. Best example is a media recorder that is scheduled to record some TV show. The room agent powers on the device just in time for its usual operation and thus ensures that a maximum amount of energy is conserved when the device is not used. The final step of the room left protocol is that the room agent sends the corresponding messages to the automation system agent that then switches off the devices. Before these commands are sent the room agent furthermore waits for a predefined time to see if the room becomes occupied again soon (i.e., the user may return soon).

Another communication relationship exists for the realization of thermal comfort in a smart home room. The scenario, as shown in Figure 4.8, starts with the user agent communicating to the room agent that the user entered the room. Alternatively, actions that establish thermal comfort can be initiated by the data management agent. This agent can be capable of predicting future occupancy, for example with the help of profiles. It can therefore tell the room agent that an occupancy of the room is predicted for a specific future point in time. The room agent starts with requesting preferences from the user agent and can also obtain a weather prediction from the data management agent. This weather prediction can later be required for specific operations that influence thermal comfort. Again, the depicted sequence diagram concentrates on a cooling example. To fulfill this cooling need, the room agent again requests the list of possible cooling scenarios from the data management

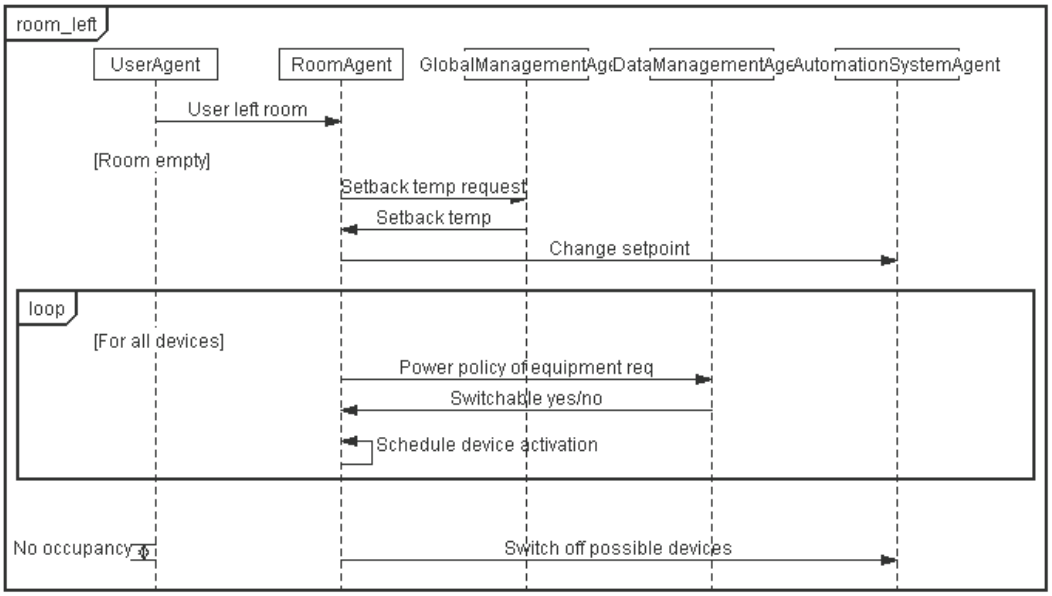

Figure 4.7: Sequence diagram – Room leave protocol

agent. It then internally deliberates on the best cooling action, taking into account for example energy efficiency considerations. In Figure 4.8, two cooling possibilities are shown. The first option uses free cooling, while the second one uses mechanical air-conditioning. In the selection process, the room agent can also consider weather or miscellaneous other information (e.g., outdoor noise or air pollution levels). This is for example necessary to detect that currently the windows cannot be opened due to heavy rain or storm. When the room agent has decided on a particular strategy, it calculates the appropriate control parameters and communicates them to the automation system agent.

Figure 4.9 illustrates the communication that is required so that energy consuming appliances can be started at the most energy efficient point in time. The protocol "appliance schedule" starts when a smart home inhabitant notifies the smart home system that an appliance is ready for operation. This message is received by the global management agent which is responsible for the scheduling task. The global management agent then queries the data management agent for information on the appliance, mainly to know how long the operation of the appliance is expected to take. Optionally (if it was not already communicated by the inhabitant), it also requests the deadline when the appliance operation needs to be finished from the data management agent. The answer is given in form of the user occupancy schedule. In

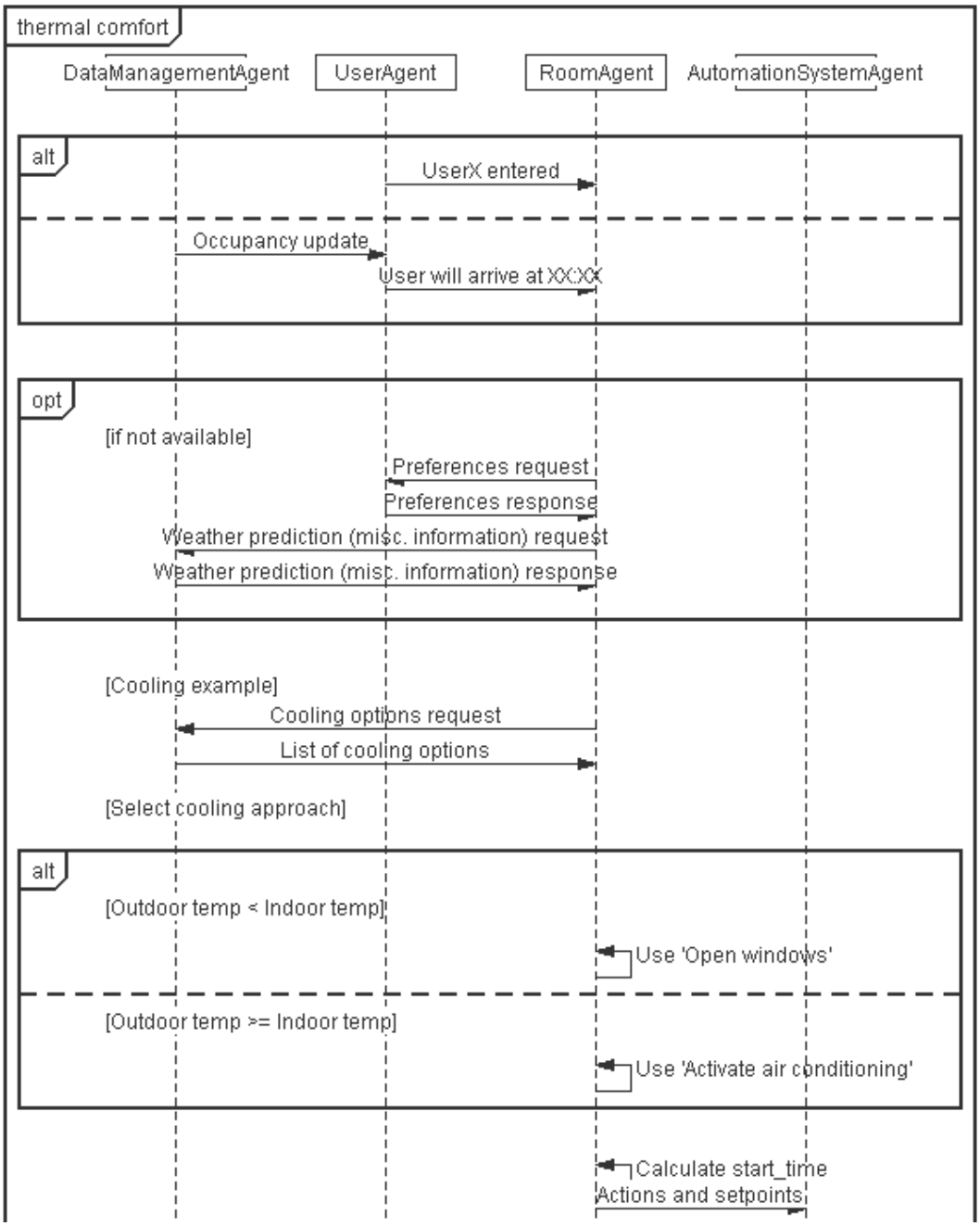

Figure 4.8: Sequence diagram – Thermal comfort protocol

a next step, the global management agent obtains the weather prediction from the data management agent. These data are used for an estimation if and when energy

from local energy producers are expected to become available. The protocol then differentiates between three cases. If no local energy is expected, the appliance start is scheduled only with respect to the user preferences. If energy might be produced locally before the deadline is reached, the appliance operation is delayed until energy becomes available. If this prediction is wrong the appliance is started in time using the standard energy provider of the smart home so that the deadline can be kept. In any case, the global management agent initiates the device operation by means of a message sent to the automation system agent.

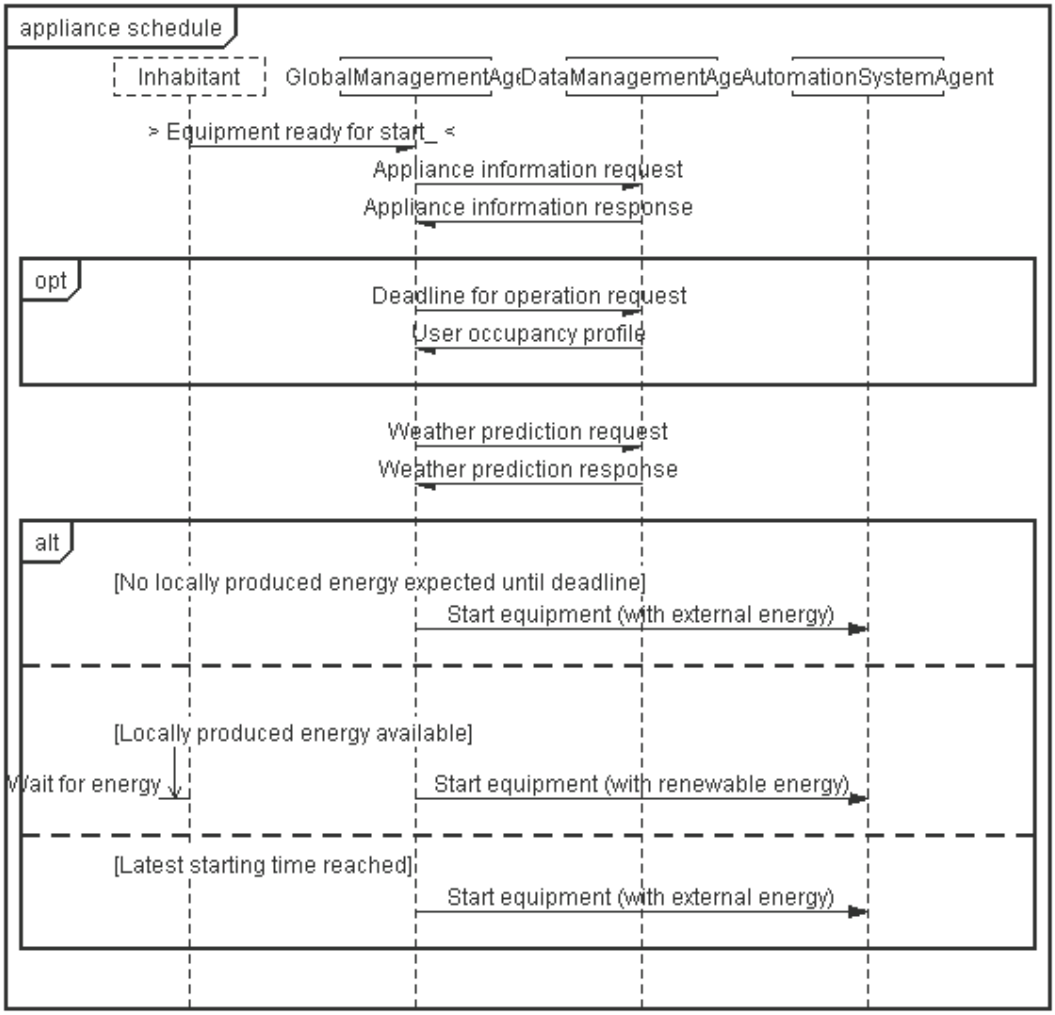

Figure 4.9: Sequence diagram – Appliance schedule protocol

The sequence diagram of the "house empty" protocol ensures that the home is set to of safe, secure and energy efficient state when the smart home is unoccupied. When a user leaves the house, the user agent notifies the global management agent of the event. The global management agent keeps track of the occupancy in the house and can thus detect of the house is empty. In this case, it sends a message to the automation system agent so that the house is first of all set into a secure state. Afterwards, the global management agent requests a list of the device states of all safety related devices (e.g., oven or water outlets) from the data management agent and analyzes the device states. If a safety problem is detected (e.g., an active oven) the device is first set to a safe state. This is accomplished by the global management agent sending a corresponding message to the automation system agent. Then, a message is sent from the global management agent to the user agent so that the agent can inform the user about the automatic action. As a third part, the protocol also ensures that unused devices in the home are switched off. For this purpose, a list of idling devices is requested from the data management agent. The global management agent then communicates to the automation system agent to separate selected devices from the power supply.

The final Prometheus artifact of the architectural design phase is the *system overview diagram*. This diagram summarizes the system architecture that has been created during this Prometheus design phase. Thus, it provides an overview of the agent types, their main percepts and actions as well as of the (communication) protocols that exist among the agents.

The system overview diagram for the smart home system is shown in Figure 4.11. There, the main identified agent types are shown in the middle. On the left side, the most important percepts and actions of the system are listed and associated with the implementing agents. On the right hand side of the agents, the communication relationships among them are depicted as protocols. The system overview furthermore illustrates which agents participate in the different communication actions.

4.1.3 Detailed Design Phase

The detailed system design phase of Prometheus is geared towards the implementation of the agent system. Hence, it concentrates on the further specification of the agent internals. For this purpose, the methodology knows *capabilities*. These capabilities are related to the (Prometheus) *functionalities* of the system specification phase and can be regarded as a more implementation related view of them. In course of their definition, the original functionalities may be broken down into sub-

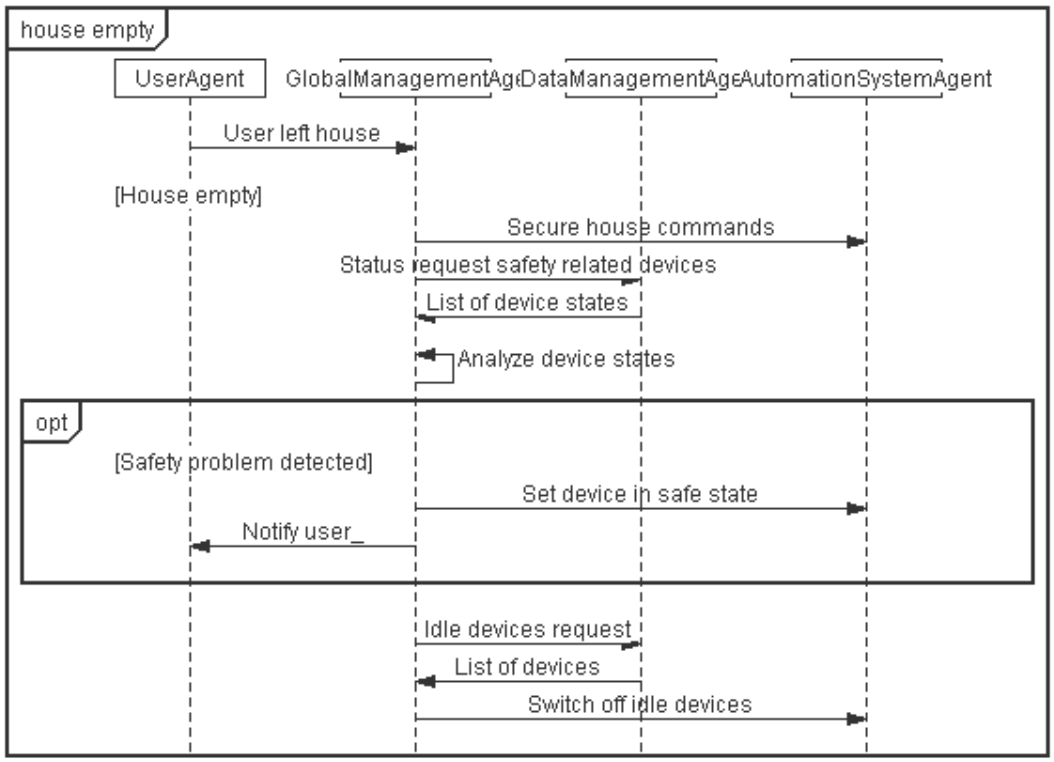

Figure 4.10: Sequence diagram – House empty protocol

parts where each part then represents a capability. Likewise, functionalities might also be merged in the detailed design phase, resulting in more complex (also more complex to implement) capabilities. The Prometheus capabilities provide a notion of which goals can be reached by the later execution of some code in the software system. Regarded from another viewpoint, capabilities provide a means to relate system goals to implemented code. They also encapsulate specific functions, which allows the reuse of pieces of software code at different places in the later system.

For a system of the complexity of the smart home system, capabilities might however not be the best choice. This is especially rooted in the fact that the smart home system offers many different control possibilities – or roads to success – to reach the system goals. This characteristic is not easily represented in a capability, mostly due to their rather straightforward way of associating plans to goals. Plans implement the goals in the agent system. In Prometheus, the standard case is that one goal is realized by (only) one plan. In the smart home system, the different control possibilities call for many different plans realizing the same goal. This obviously

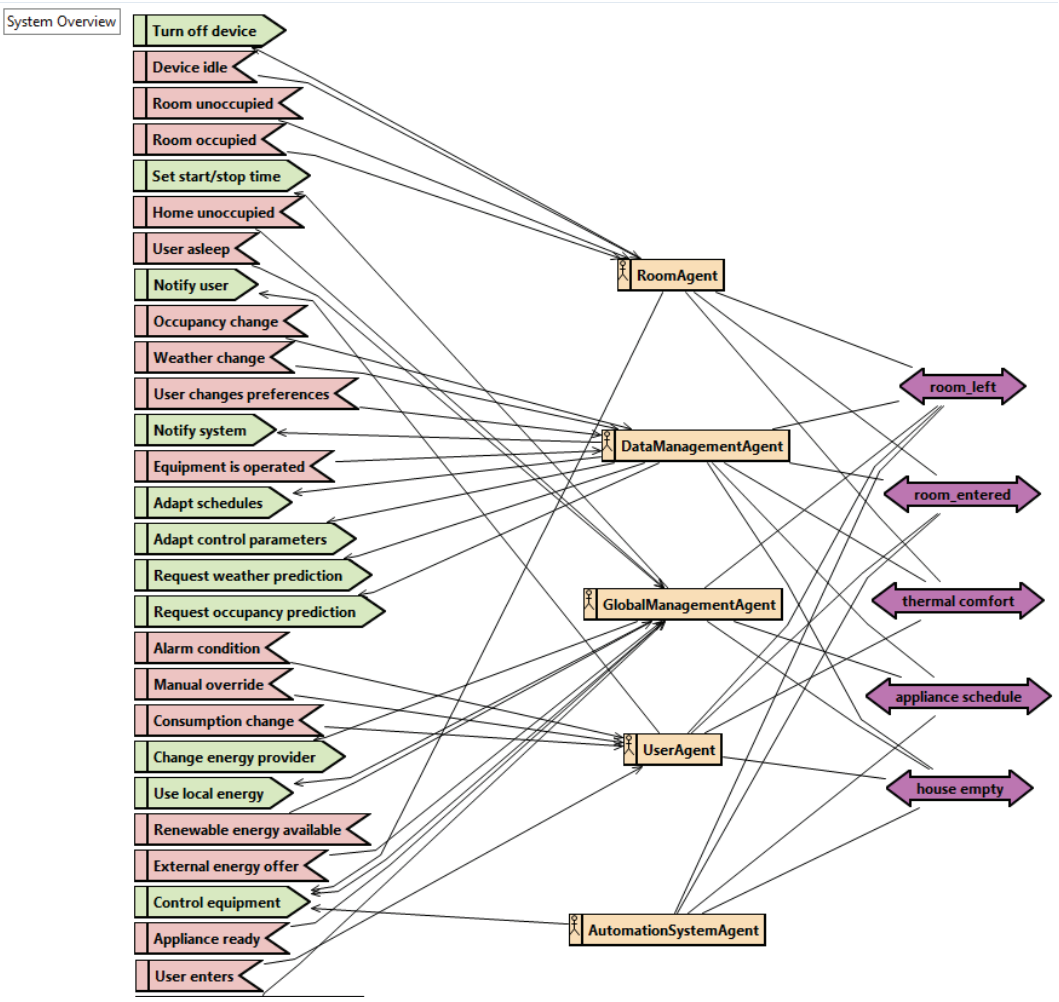

Figure 4.11: Prometheus system overview diagram

increases the system complexity but is a mandatory characteristic for the operation of a smart home.

Although the Prometheus methodology has a high coverage of the conceptual steps of the system design and specification, it offers comparably less support regarding the implementation, especially for managing the complexity of the implementation. In particular, it does not provide any means to decrease goal and plan complexity, but follows a strategy where goals remain rather superficial. Given the goals that were identified during the system design phase of Prometheus, this would mean that plans for goals of the complexity of "ensure visual comfort" would have to be written. However, such a goal encompasses several aspects which would then

have to be aligned in one Java method only. This not only increases programming complexity and with it the error proneness, but also limits the flexibility of the system. The agent system's intelligence is to a considerable share attributable to the intelligent activation of goals and the according selection of appropriate plans that achieve the goals. Thus, if there are only rather global goals such as is the case in Prometheus, this means that the goal activation part cannot contribute to significant intelligence.

Summarizing, it can be said that Prometheus – at the detailed design phase – does not really harmonize with the requirements of a smart home system, in particular its typical broad range of possibilities that exist to achieve the initial goals. For this reason, the Prometheus methodology is no longer (from the detailed design phase on) strictly followed in this dissertation. For the smart home system, a more detailed representation of goals and plans is identified as mandatory criterion for a successful (i.e., easier and less error prone) implementation. It comes to help that research has developed methods that help the system designer and engineer to cope with this problem. In particular, other agent design methodologies include related steps that can provide input for the smart home case. As will be explained in the following paragraphs, these approaches are very valuable for further specification of the implementation related details of systems of the complexity similar to the one of the smart home.

A common step to reduce the complexity of such a highly adaptive system such as the smart home system is hierarchical goal decomposition. This method is originally known from the artificial intelligence and planning domains (e.g., as Hierarchical Task Networks) [61], but can also be well applied to the smart home case. This view is underlined by the goal overview diagram obtained by applying the Prometheus methodology (cf. Figure 4.1), which presents a first hierarchical goal decomposition of the system. In this case, the goals are sequentially broken down from top-level goals into smaller, more concrete (sub)goals that can be achieved and described more easily. However, this view does neither incorporate any details nor any information on how these goals will be implemented later.

For this purpose, a *goal-plan hierarchy* is useful. The goal-plan hierarchy is similar to the hierarchical goal decomposition but additionally considers implementation aspects. These implementation aspects are represented as plans. The plans describe actions that are executed to fulfill a goal of the next higher hierarchy level and are inserted in between the different goal hierarchy levels. Plans that are at the very bottom level (i.e., those that do not have any subgoal below them) are atomic in the

sense that there the concrete actions that are executed by the system are modeled. Thus, the goal-plan representation is in fact quite implementation related in a way that it breaks down and specifies the functionalities that later have to be implemented as source code in a detailed way. This also reduces the effort for the programmer as the single plans that need to be implemented are less complex, although the overall system functionality and complexity remain the same.

There are several further benefits when additionally considering plans at this late design stage. One motivation is that they visualize the system's robustness and flexibility. For each goal, the agent system can potentially choose from a range of plans to reach this particular goal. Thus, a goal not necessarily fails only because one particular plan fails. Furthermore, the structure that is imposed on the agent system helps the system developer to exploit natural hierarchies to reduce the problem complexity. At the same time, the software implementation becomes easier as atomic actions are clearly identified and can thus be implemented more easily compared to more complex constructs, such as the capabilities of Prometheus. In practice, most functionality is implemented in the atomic plans, where – thanks to the hierarchical decomposition – only comparably simple functions (yet a higher number of them) need to be written in the programming language. Another point is that identified plans may be reused in other scenarios, in particular if they implement standard functionalities.

With respect to the implementation of the system functionality in the agents, a main benefit of the goal-plan hierarchy is its close relation to agent internals of agents that follow the BDI paradigm. In particular, the notion of plans in this case is fully in line with their meaning in the implementation of BDI agents. As described in Section 3.2.2, plans of BDI agents are a particular set of actions that the agents can execute to bring about (one of) the desired world state(s) (i.e., to realize one of its goals).

This similarity will be exploited in the detailed design phase of the smart home system. Although the goal-plan hierarchy is not part of the Prometheus methodology, it is partly related to the capabilities of Prometheus, which can be considered as more complex or aggregated plans. Furthermore, the percepts and actions that were identified in Prometheus can also be used in the goal-plan representation, where they are implemented in the plans. Another argument that supports the decision to follow this decomposition step is that the use of the goal-plan approach is not new to the design of agent based systems. A very similar design artifact for example exists in the Tropos methodology with the goal decomposition diagram (cf. Section 3.4). The use

of the goal-plan hierarchy or closely related approaches is also recommended in many research works, for example in a paper by Braubach et al. on goal representation in BDI agent systems [116], for hierarchical planning as in [117] or as goal decomposition tree such as in [118].

In Figure 4.12, a goal-plan hierarchy for the thermal comfort use case of the smart home is shown. Starting from the top-level goal to build a smart home system that (also) *assures thermal comfort*, a hierarchical goal decomposition is performed to identify all subgoals as well as all plans that fulfill these goals. In the diagram, goals at the same hierarchy level are always connected with `AND` semantics, while plans of the same level represent alternative options, i.e. they are `OR` connected. At the leaves of the tree, atomic (set of actions) are represented, where atomicity relates to a particular context.

The goal-plan hierarchy is constructed following two approaches, a bottom-up and a top-down one. It starts from the goal overview created in course of the application of the Prometheus methodology (cf. Figure 4.1) and refines as well as extends this design artifact. In the top-down approach, the system goals are decomposed in sub-functionalities, where each functionality is again associated with a goal. This step requires good domain knowledge of the system under design and is thus executed by (or with the help of) a domain expert. An example can be seen in the presented goal-plan hierarchy. Within it, for example, the domain knowledge that different kinds of control strategies for the room temperature control exist, are encoded.

Following the bottom-up view, an aggregation of the system percepts and actions into more complex units is performed. This step is similar to the definition of Prometheus capabilities, but in the present case it always strives for an alignment with the top-down view. Still, the actions and percepts are closely related to the ones that were already identified in the Prometheus system overview diagram (cf. Figure 4.11). Hence, the creation of the goal-plan hierarchy is a process that has to happen iteratively, where during each iteration an alignment of bottom-up and top-down view is required. Although this approach can be challenging, it nevertheless ensures that neither domain knowledge aspects nor implementation details are neglected. Thus, it realizes a trade-off between them.

For the thermal comfort case that is depicted in Figure 4.12, this means that the overall goal is to ensure thermal comfort. The only available plan to achieve this goal is the *thermal comfort plan* which requires three new goals (that model more specific requirement) to be fulfilled, *air humidity*, *temperature* and *air quality*. To control the room temperature, a smart home system can have several different means to select

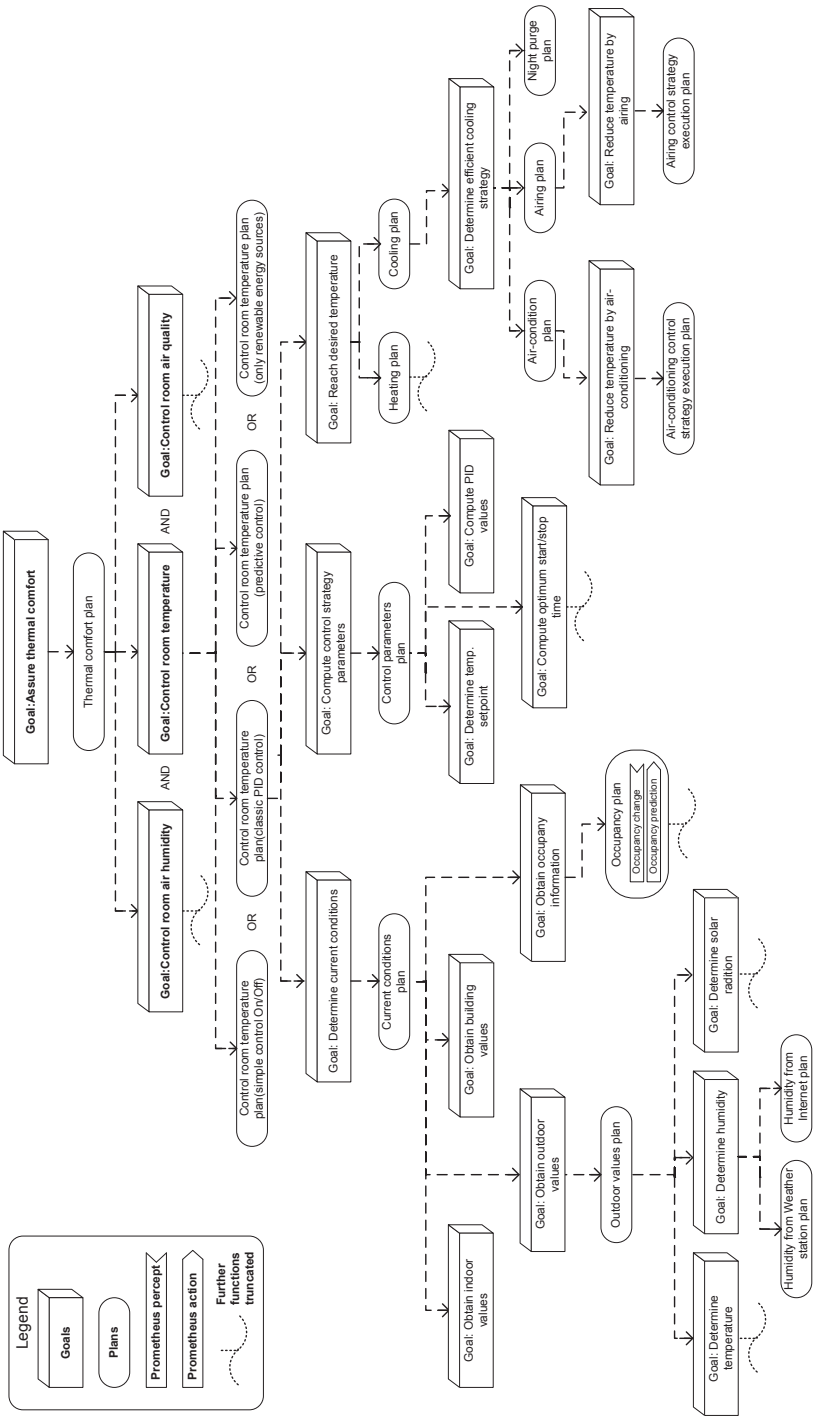

Figure 4.12: Goal-plan hierarchy diagram of the thermal comfort use case

from. These are modeled as the four *control temperature plans*, allowing the system to choose among an on/off, classic PID, a predictive controller or other strategies. In BDI agent systems, the selection process of one particular plan is accomplished by the execution of means-end reasoning mechanisms in the agent platform. The system designer can influence this process, for example, by assigning priorities to plans. In a similar approach, additional knowledge in the system (e.g., information stored in the knowledge base) can be exploited to infer the currently most appropriate plan. This way, for example energy-efficient strategies can be preferred in certain cases over simple plans that are characterized by their robust behavior.

In Figure 4.12, only the example of a PID based temperature control is worked out in detail, all other control options are not fully elaborated to keep the diagram complexity within reasonable limits. As can be seen in the figure, very basic (on/off) control as well as sophisticated control strategies (predictive control) a represented and may be used by the system. The reason for this is that the smart home system shall be generic enough to operate on a rudimentary amount of automation equipment. If there is no dedicated temperature controller present, an on/off strategy can be used instead. Likewise, if more (sensor) data and/or more and better equipment is available, advanced strategies are implemented. Nevertheless, also the PID controller plan as elaborated in the figure can be considered as quite novel as it might incorporates miscellaneous additional parameters (e.g., the *building structure*, *occupancy information* or *solar radiation*). In particular, the depicted thermal comfort control approach combines two advantages: first, this strategy only demands a classic PID controller to be available in the automation system. Secondly, both the control performance and resource efficiency are greatly enhanced by the additional information that is available, for example, by the computation of an optimum start/stop time based on building physics, building layout and structure parameters.

At the bottom of the goal-plan hierarchy, the most detailed plans can be found. These plans represent atomic actions for the given use case. It are in particular these plans that are implemented by the system developer in the agent framework as functional code. In the presented example, functions that allow to obtain information from the environment (e.g., *humidity from Internet*) or that implement actions to be taken in the environment (e.g., *air-condition control strategy execution*) need to be implemented. Furthermore, an example how percepts and actions relate to plans is shown in the *occupancy plan*. Within this plan, the main action as well as the main percept of this plan (that were already captured in the Prometheus system overview diagram, cf. Figure 3.22) are also depicted. *Occupancy change* is a percept

of this plan, meaning that the plan has information on such an event available. This perceived information can then be used to implement some functionality, in the presented case, the prediction of occupancy. Such a prediction is calculated in the plan and made available to other agents, i.e. the agent implements an action (*occupancy prediction*) that makes the information available.

The goal-plan hierarchy is thus already very related to the implementation of the agents. It provides the goals and subgoals as well as different realization scenarios for the goals by means of plans. Additionally, it already captures some aspects of the later plan implementation, in particular which percepts and actions are needed in the plan. The full goal-plan tree can be used to directly determine the goals that must be encoded within the agents as well as the plans they must implement. All implementation related details are presented in the next section.

4.2 Prototype Implementation

The final decision to be made is the selection of a technology for the implementation of the smart home system concept. The implementation related questions as well as a prototype implementation of the proposed smart home system are presented in this section.

At the end of the detailed design phase, Prometheus provides an approach that supports the transformation of the design artifacts into concepts of the JACK agent architecture [119, 120]. This step is obviously very implementation related and furthermore JACK technology specific. However, for the smart home system, JACK does not constitute the first choice for an implementation of the agent system. Main reason is the fact that JACK is a commercial product that does not offer a free implementation platform. If JACK is used, considerable license fees apply. Furthermore, the implementation of the JACK framework is not openly available so that its internals and architecture decisions can neither be evaluated nor changed.

Although Prometheus offers exclusive implementation support (e.g., automatic generation of source code) for JACK, the implementation of agents that were specified in Prometheus is not limited to the JACK framework. Rather, as the specification obtained from following the Prometheus methodology is generic enough, agents can be implemented in most common BDI agent frameworks. The most common frameworks that can be used to put a BDI agent system into practice are PRS [75] which was one of the first BDI frameworks, JAM [76] which focuses on data mining support and Jadex [121] which is based on the well-known JADE platform [122].

4.2.1 Jadex

In order to facilitate the realization of the smart home system, an implementation that includes the most important BDI architecture components such as belief base, reasoning and (re-)action parts is desired. Regarding these criteria, Jadex is considered to be a well suited incarnation of such a BDI implementation framework. Jadex (which stands for *JADE eXtension*) provides a hybrid (reactive and deliberative) agent architecture that is capable of representing mental states in JADE agents. It thus extends JADE agents in the direction of BDI components but at the same time relies on the proven middleware approach of JADE. This middleware provides extensive support for the general agent functions. Most important, it manages all communication and runtime environment issues, for example, by providing FIPA (Foundation for Intelligent Physical Agents) compliant communication services to the agents. Furthermore, it provides support for agent management and implements an agent/service directory. The relation between JADE and Jadex is shown in Figure 4.13.

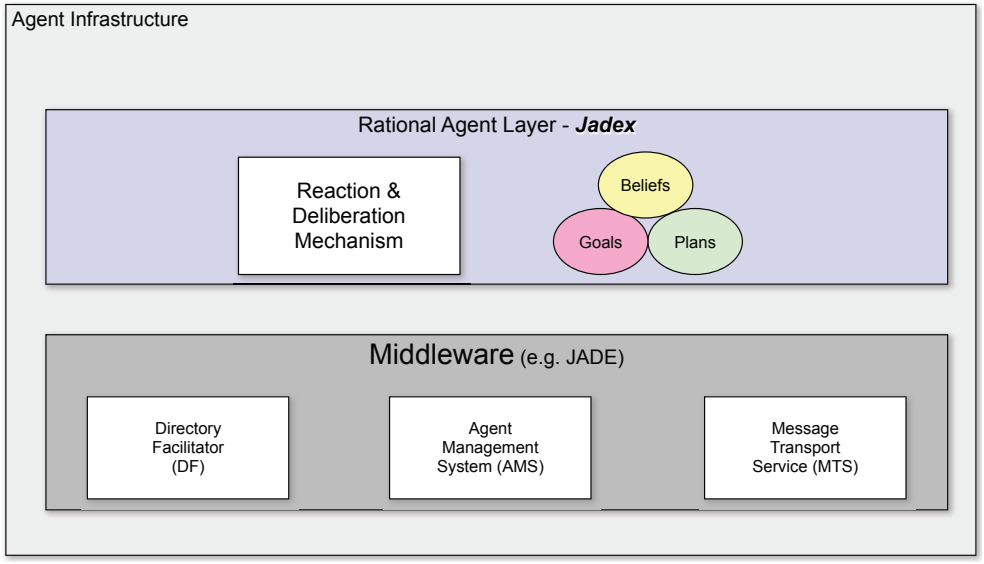

Figure 4.13: Relationship between JADE and Jadex [123]

One advantage of Jadex is that Jadex agents implement all important BDI agents' internal components such as an agent's mental states, beliefs, goals and plans. Furthermore, Jadex allows agents to maintain communication relationships hence multi-agent systems can be implemented with it. The communication services are provided

transparently by the underlying JADE platform. This is also the case concerning the interaction with the environment, i.e., sensing and acting.

Although some adaptations and extensions to the original BDI concept as presented by Rao and Georgeff [71] were made, Jadex agents remain fully compliant to the BDI model. The chances were made mainly because of implementation reasons as well as to increase the simplicity and performance of Jadex agents. Compared to the original BDI concept, adaptations mainly concern extensions to this model, such as the added support for a more complex deliberation process.

In Jadex, beliefs are stored within every agent in a *belief base* that represents the agent's knowledge. The exact semantics of the knowledge base are not specified, but since JADE and Jadex are written in Java, encapsuling the information in Java objects is advisable. However, as proposed in this dissertation, also a knowledge base implemented as an ontology can be used to represent the knowledge connected to a smart home. Although the ontology has to be considered as an external subsystem (regarding the agent system), the knowledge stored there can still be provided to the agents via Java objects [124].

However, unlike specified in the original concept of Rao and Georgeff, Jadex does not explicitly use the BDI concepts *desire* and *intention*. In Jadex, desires are represented as goals, with different goal types existing. Goals may specify a particular environment state that shall be brought about, some state that shall be kept or only a set of miscellaneous actions that shall be executed when the goal becomes active. Control over goals and their execution is gained by a broad range of additional goal properties that can be assigned to the goals. Goals are activated (or deactivated) based on the outcomes of the *deliberation* process that is provided by and executed in the Jadex architecture.

Finally, the actions that can be executed by an agent are modeled as plans. Relating to the BDI nomenclature, intentions are plans that the agent has committed to execute. Thus, plans are used to achieve the goals specified within the agent. In Jadex, different plans may exist for each goal, where all plans are applicable to achieve the goal. For example, in the smart home domain, if the goal is to reach a higher temperature in a room, plans may be to open the radiator valves or the sun blinds, respectively. For each plan, preconditions can be specified under which the plan is applicable.

The decision which plan is executed is accomplished automatically by the Jadex reasoning engine, considering the preconditions and additionally following for example, random or priority schemes. This *means-end reasoning* process basically takes

the committed goal, the current state of the environment and all associated plans as inputs and identifies one plan that can be executed to achieve the goal [123].

Plans contain the atomic elements of an agent, i.e. the actions that can be executed by the agent. In Jadex, plans are implemented as arbitrary Java code. Thus, actions may sense or alter the environment, send messages to other agents or objects and also trigger internal events in an agent, e.g., change the belief base or activate other goals. Especially this latter point is of importance as it closely corresponds to the goal-plan hierarchy approach. In both cases (higher level) goals are fulfilled by plans that in course of their execution may activate subgoals. These subgoals are again realized by other plans that themselves may trigger subgoals and so forth.

The abstract architecture of Jadex is shown in Figure 4.14. It can be seen that the BDI components constitute the agent's internals and that Jadex follows an event based scheme where any internal or external action is treated as event.

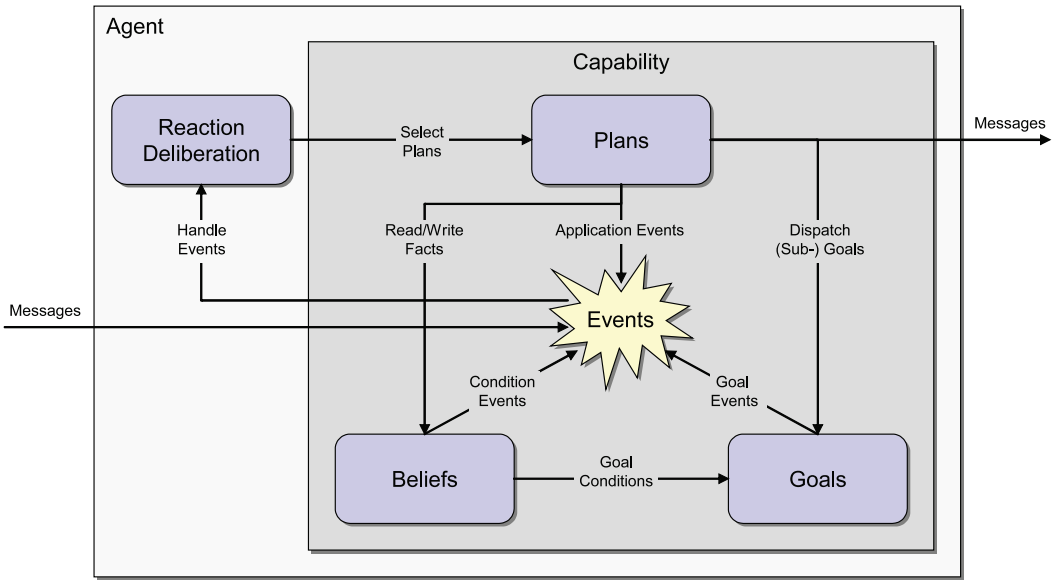

Figure 4.14: Abstract architecture of Jadex [125]

For the implementation of the smart home system in Jadex, the goal-plan hierarchy artifacts are used as starting point. From them, it is possible to directly determine the goals that must be encoded within the agents as well as the plans they must implement. Although this process at first may seem completely straightforward, an implementation in Jadex requires one more step to be accomplished. One

particular characteristic of Jadex is that it defines four different goal types, where each goal type models a particular behavior in the Jadex system [126]. These goal types can be used to define further goal characteristics that influence for example the lifetime of a goal or its temporal behavior [116]. The four goal types of Jadex are the *achieve, maintain, perform and query* goals.

The *achieve goal* works towards some desired world state that is included in the goal description, but does not specify how to reach it. The system can thus try different alternative ways to achieve the goal. A *maintain goal* is used to maintain some specified world state. In this case, the agent keeps track of the world state, detects any deviation from it and continuously executes associated plans to re-establish this state. A goal that directly encodes (only) some actions to be executed is defined as a *perform goal*. Finally, a *query goal* is used to obtain some information the agent needs.

As these goal types are obviously not explicitly captured in the goal-plan hierarchy, they must be mapped manually. While the lack of an automated goal type assignment may be considered a drawback, this shortcoming is alleviated by the fact that in most cases only one mapping possibility is reasonable. In practice, a good mapping can be accomplished by following simple rules of thumb. For example, in any case where information shall be obtained (from the environment, from other agents or other miscellaneous sources...), the query goal type can be used as it provides respective mechanisms to execute these tasks. Similar, whenever actions shall be executed in the environment and no feedback for these actions is required, a perform goal should be applied. In the smart home, this primarily concerns all functions related to safety, for example an emergency stop of the ventilation equipment. Furthermore, achieve goals are applicable when some change in the environment is targeted (e.g., the lights shall be turned on). This goal type is most common during the regular operation of the smart home system. In certain cases and depending on the application scenario as well as the concrete implementation, a maintain goal is an alternative to the achieve goal. It is applicable when an aspect of the environment shall be preserved (e.g., the humidity level shall be kept at a constant value).

Although different goal types are known by Jadex, their practical use in a system is sometimes limited. In practice, all goals could, for example, be modeled as achieve goals. In that case, the use of specific target conditions together with a corresponding plan implementation that checks these conditions can be an easier and more flexible way of representing goal types than their explicit consideration already in the agent definition file. Hence, the programming approach has an influence on the usefulness

of the different goal types. Therefore, depending on the system it can be a reasonable approach to model all goals as achieve goal and implement the more specific features in the corresponding plans.

Finally, in Jadex one or more goals are implemented by a single agent. The agents are described in an XML based agent template called the Agent Definition File (or ADF). From this ADF, one or more such agents can be instantiated at runtime. In the ADF file, the agent's beliefs and goals are modeled as well as the plan heads are defined. The plan bodies (i.e. the implementation of the plan actions) are not part of the ADF, but are written as standard Java classes that contain arbitrary Java code. Once these steps are accomplished, the agents can be run on one or more, possibly distributed agent platforms and the operation of the smart home system may be started. The resulting architecture of a Jadex agent is shown in Figure 4.15.

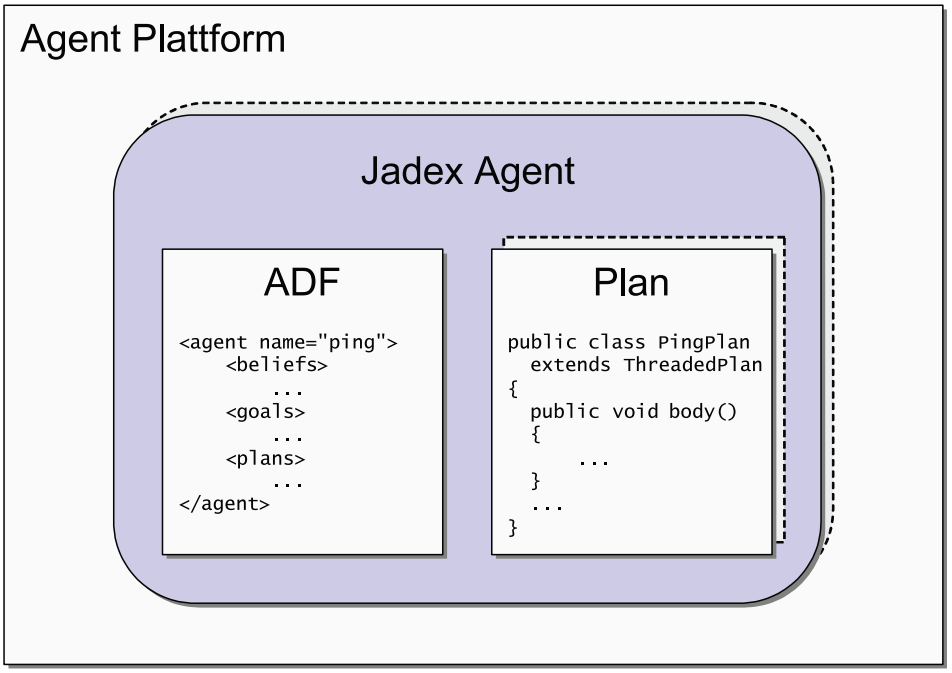

Figure 4.15: Jadex agent [121]

4.2.2 Implementation

For the prototype implementation, a standard office computer was used. There, the latest Jadex release version 2.2.1 was used in combination with the Java SE Development Kit 7, Update 9 (version JDK 7u9).

In the Jadex framework, the agent system is initialized with the help of an XML file termed `application.xml` that describes the system parameters. This XML file is loaded and interpreted by the Jadex engine. For the smart home system, this XML file is shown in Listing 4.2. In the listing, several lines of code were omitted, but its general structure was preserved. The deleted parts are marked with three dots ("..."). For example, in the `imports` list several (less important) imports are missing.

The XML file begins with a list of `imports` that are used to include both functions from Jadex as well as from the Java environment. In the section `component types`, all agents that are used in this particular multiagent application are initialized. Thus, the five identified agent types are listed and their ADF file is referenced (e.g., `globalview.agent.xml` refers to the ADF file of the global management agent). Practically, any Jadex agent can be included here, hence agents can be reused in different applications. The `configurations` part is used to define different start up conditions of the system. At the system initialization, one configuration can be selected and agent instances are created accordingly. Thus, different configurations can be created to accommodate different conditions, for example, different building structures.

In the provided example, the "easy rooms" configuration ensures that at start up a global management agent named "global" is created. Furthermore, the configuration is used to pass parameter values to the agents, where the type and range of such parameters is virtually unlimited. As shown, the global management agent is already aware that three different rooms ("kitchen", "living room" and "entrance") exist in the smart home. Likewise, user agents and room agents are created and initialized with default values. It is important to note that these initial values are not static but only represent start up values that can be changed during runtime. This is also the case for the agents. Additional agents can be launched during the system operation as well as agents can be dismissed again.

```
<?xml version="1.0" encoding="UTF-8"?>
<componenttype xmlns="http://jadex.sourceforge.net/jadex"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://jadex.sourceforge.net/jadex
```

```
http://jadex.sourceforge.net/jadex-component-2.0.xsd"
  name="smarthomesystem" package="smarthome">

  <imports>
    <import>jadex.commons.Tuple</import>
    <import>jadex.commons.*</import>
    <import>jadex.planlib.*</import>
    <import>jadex.bridge.fipa.*</import>
        ...
  </imports>

  <componenttypes>
    <componenttype name="roomagent" filename="RoomAgent/room.agent.xml"
        />
    <componenttype name="globalmanagementagent" filename="
        GlobalmanagementAgent/globalview.agent.xml"/>
    <componenttype name="automationsystemagent" filename="
        AutomationsystemAgent/bas.agent.xml"/>
    <componenttype name="datamanagementagent" filename="
        DatamanagementAgent/dme.agent.xml"/>
    <componenttype name="useragent" filename="UserAgent/user.agent.xml"
        />
  </componenttypes>

  <configurations>
    <configuration name="Easy rooms">
      <components>
        <component name="global" type="globalmanagementagent">
          <arguments>
            <argument name="mainroom">"kitchen"</argument>
            <argument name="roomlist">
              new Tuple[]
              {
                new Tuple("kitchen", "no", "20"),
                new Tuple("livingroom", "no", "20"),
                new Tuple("entrance", "no", "20")
              }
            </argument>
          </arguments>
        </component>
        <component name="Wolfgang" type="useragent">
          <arguments>
            <argument name="username">"Wolfgang"</argument>
            <argument name="Temp">22</argument>
```

```
            <argument name="Light">20</argument>
          </arguments>
        </component>
        <component name="Ruth" type="useragent">
          <arguments>
            <argument name="username">"Ruth"</argument>
            <argument name="Temp">25</argument>
            <argument name="Light">29</argument>
            <argument name="MMpreference">"on"</argument>
          </arguments>
        </component>
        <component name="livingroom" type="roomagent">
          <arguments>
            <argument name="roomname">"livingroom"</argument>
            <argument name="hasblinds">"true"</argument>
          </arguments>
        </component>
        <component name="kitchen" type="roomagent">
          <arguments>
            <argument name="roomname">"kitchen"</argument>
          </arguments>
        </component>
        ...
        <component name="entrance" type="roomagent">
          <arguments>
            <argument name="roomname">"entrance"</argument>
          </arguments>
        </component>
      </components>
    </configuration>
  </configurations>
</componenttype>
```

Listing 4.2: Smart home application XML

Apart from this system initialization, also the agent types must be encoded in the Jadex system. This is done with the ADF file which contains all details on the agents. When a new agent of a particular type shall be added in the system, Jadex internally creates a model based on the details specified in the XML file. The ADF hence provides a template for an agent type. As an example, the ADF of the smart home room agent is provided in Listing 4.3. The ADF XML was edited to show only the main parts. Additionally also the attributes of the main parts were truncated to increase clarity.

The main components of the agent description are `goals`, `beliefs` and `plans`. As shown in the listing, the agent XML file again starts with an `imports` section. Then, `goals` are specified. In the example, three (initial) goals are shown that illustrate how an agent registers at the Jadex directory service. For each goal, the goal type is encoded so that the agent discovers the agent registry (`achievegoal` "df_search") and keeps its registration (`maintaingoal` "df_keep_registered"). However, not all goals need to be encoded in XML. In case of the room agent, many goals are added only when a user enters the room.

In the next section, the agent's initial beliefs are specified. The example shows a set of beliefs (`beliefset` "userregister") which groups beliefs that represent the user's desired values in the room. During runtime, the initial values are overwritten with the current user values when the user enters the room. For the smart home, a long list of beliefs exists of which some are shown in the listing. For example, the room agent knows the current temperature (`actemp`), the desired temperature (`temp`), the state of the lights (`lighton`) and further facts, such as the get up time of a user (`getuptime`).

The last main part of the ADF specification are the agent plans. For each plan only the plan head is defined in the ADF, but the plan implementation is managed separately. In each plan head, the attribute `body class` refers to the implementation file name (i.e., the piece of Java code that implements the plan). Additionally, for each plan a `trigger` exists. This attribute defines when a plan is executed. The trigger provides the programmer with several possibilities to control the selection (and execution) of a plan. In the basic case that is also depicted in the listing, the trigger can be used to check for a matching message type. Similar, also a matching of values can be implemented. A more advanced possibility is to use arbitrary boolean expressions to check whether certain beliefs currently evaluate to true or not. These beliefs could also be rooted in the knowledge base of the smart home and in that way represent more complex conditions. Furthermore, additional attributes that control the plan execution exist. Among them, the pre- and context conditions are most important. Pre-conditions are evaluated *before* a plan is selected for execution and can thus be used to control whether the plan becomes applicable in the first place. A context condition describes some condition that is evaluated *during* the plan execution, and – if it is violated – results in the plan being aborted.

In the example, the room agent has a plan "entered room" which is triggered by a message event (`messageevent` "request_enterroom"). The Java code of the plan is found in a separate Java class with the name "EnteredRoomPlan".

Apart from the BDI related concepts, Jadex also specifies further details of the message events that were just used as trigger for the plans. For this purpose, an attribute (`messagevent`) exists in the ADF. Every incoming or outgoing message (indicated by the `direction` parameter) of the agent is defined in this section. In the smart home system, only messages of type "FIPA" are used, meaning that the agent communication is fully compliant to the FIPA communication standard. The FIPA standard defines a set of performatives (e.g., `request`, `inform`, etc.) and the meaning of the performatives. However, the standard makes no assumption about the message content which is left completely open to the implementation of a concrete system. During runtime, Jadex compares all incoming messages with the locally known message types of the agent. To support this matching process, a `match` attribute exists in the ADF that is used to specify some condition that uniquely identifies this particular message. In other words, it helps to select the best fitting `messageevent` among all specified ones whenever a new message is received.

In the example, the agent waits for the reception of different messages, e.g., the "request_enterroom" message which is of type "Fipa.REQUEST". The message event is detected following the `match` condition. Only if a received message has a content that starts with the String "enter_room", the room agent internally triggers a corresponding `messageevent` (the "request_enterroom" event). Clearly, all message events that were specified in the plan section need to be further defined in the `event` part of the ADF.

```
<?xml version="1.0" encoding="UTF-8"?>
<agent xmlns="http://jadex.sourceforge.net/jadex"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://jadex.sourceforge.net/jadex
http://jadex.sourceforge.net/jadex-bdi-2.0.xsd"
  name="room"
  package="smarthome.RoomAgent">

  <imports>
    <import>jadex.planlib.*</import>
    <import>jadex.bridge.fipa.*</import>
    <import>java.util.logging.*</import>
    ...
  </imports>
    ...
  <goals>
    <!-- Make the keep registered goal available. -->
    <maintaingoalref name="df_keep_registered">
```

```
      <concrete ref="dfcap.df_keep_registered"/>
    </maintaingoalref>
    <!-- Include request goal type from dfcap. -->
    <achievegoalref name="rp_initiate">
      <concrete ref="procap.rp_initiate"/>
    </achievegoalref>
    <!-- Include df search goal type from dfcap. -->
    <achievegoalref name="df_search">
      <concrete ref="dfcap.df_search"/>
    </achievegoalref>
        ...
  </goals>

  <beliefs>
    <!-- This beliefset contains the actual user values of the room. --
        >
    <beliefset name="userregister" class="Tuple" exported="true"/>

    <belief name="temp" class="int"><fact>22</fact></belief>
    <belief name="lightvalue" class="int"><fact>20</fact></belief>
    <belief name="aclight" class="int"><fact>20</fact></belief>
    <belief name="actemp" class="int"><fact>20</fact></belief>
    <belief name="lighton" class="boolean"><fact>"false"</fact></belief
        >
    <belief name="sleeptime" class="int"><fact>75600</fact></belief>
    <belief name="sleeptemp" class="int"><fact>20</fact></belief>
        ...
  </beliefs>

  <plans>
    <!-- Passive plan for entering the room. -->
    <plan name="entered_room">
      <body class="EnteredRoomPlan"/>
      <trigger><messageevent ref="request_enterroom"/></trigger>
    </plan>
    <!-- Passive plan for leaving a room, when a request leave room
        message is received. -->
    <plan name="leaves_room">
      <body class="LeaveRoomPlan"/>
      <trigger><messageevent ref="request_leaveroom"/></trigger>
    </plan>
    <!-- Passive plan for reacting, when a request tempchange message
        is received. -->
    <plan name="temperature_change">
```

```
      <body class="RoomTempChangePlan"/>
      <trigger><messageevent ref="request_tempchange"/></trigger>
    </plan>
    <!-- Passive plan for reacting, when a request prepare message is
        received. -->
    <plan name="prepare">
      <body class="RoomPreparePlan"/>
      <trigger><messageevent ref="request_prepare"/></trigger>
    </plan>
        ...
  </plans>

<events>
  <!-- Specifies an enter room request. All messages with
      performative request and start with a specific string "
      enter_room␣agentname␣lightvalue␣temperature␣mmpref". -->
  <messageevent name="request_enterroom" direction="receive" type="
      fipa">
    <parameter name="performative" class="String" direction="fixed">
      <value>SFipa.REQUEST</value>
    </parameter>
    <match>$content instanceof String &amp;amp;amp; ((String)$content).
        startsWith("enter_room")</match>
  </messageevent>
  <!-- Specifies a message, that tells a room, that the temperature
      has changed and start with a specific string "tempchange␣temp".
      -->
  <messageevent name="request_tempchange" direction="receive" type="
      fipa">
    <parameter name="performative" class="String" direction="fixed">
      <value>SFipa.REQUEST</value>
    </parameter>
    <match>$content instanceof String &amp;amp;amp; ((String)$content).
        startsWith("tempchange")</match>
  </messageevent>
  <!-- The answer message after success. -->
  <messageevent name="inform" direction="send" type="fipa">
    <parameter name="performative" class="String" direction="fixed">
      <value>SFipa.INFORM</value>
    </parameter>
  </messageevent>
        ...
</events>
```

```
</agent>
```

Listing 4.3: Agent definition file of the room agent

For all plans that were defined in the different agent definition files (i.e., the plan heads), a Java implementation (i.e., the plan bodies) must be provided. This Java code implements the agent functionalities and enables the agent to interact with and act in the environment. Standard plans extend the `jadex.bdi.runtime.Plan` class, where the main code parts are placed in the `body()` method. The plan implementation itself is not specified by Jadex. However, the Jadex API offers several interfaces that support the implementation of BDI agent functionalities.

In the following paragraphs, the "EnteredRoomPlan" of the room agent is used as a running example. The main aspects of the plan will be highlighted as code snippets and explained.

```
public void body()
{
  String  reply;
  String  cont;
  StringTokenizer stok = new StringTokenizer((String)((IMessageEvent)
      getReason()).getParameter(SFipa.CONTENT).getValue(), "␣");

  /* read message */
  if(stok.countTokens()==5){
  String planname = stok.nextToken();
  String usern = stok.nextToken();
  String tempv = stok.nextToken();
  String lightv = stok.nextToken();
  String mprev = stok.nextToken();
      ...
```

Listing 4.4: Code snippet (1) of EnteredRoomPlan

As defined in the plan head, the plan gets active when a "user enters" message is received by the room agent. Upon the reception, the room agent invokes the `body()` method of the corresponding plan. At the beginning, the message is parsed to identify the parameters that were transmitted within the message. For the present scenario, the parameters that are transmitted from the user agent to the room agent are the user name of the user that entered the room as well as the preference values for temperature, light intensity and multimedia equipment (cf. Figure 4.6). The

code for the message reception and for parsing the parameters is shown in Listing 4.4.

```
/* User already in the room? */
  String reg = (String)query_user.execute("$username", usern);
  if(reg != null)
  {
    /* if yes: do nothing */
    //println("user already in room: "+usern);
    cont = usern+"␣bereits␣registriert";
    reply = "inform";
  } else {
    /* if no: register user in the room */
    //println("user is being registered in room: "+usern);
    getBeliefbase().getBeliefSet("userregister").addFact(new Tuple(
        usern, tempv, lightv));
```

Listing 4.5: Code snippet (2) of EnteredRoomPlan

Listing 4.5 shows the next step of the plan, which checks if this particular user (identified by the username) is already registered in this room or not. In the first case, the plan execution is stopped and an error message is sent to the user agent. The Strings `cont` and `reply` are used to create the response message content and to set the *FIPA Performative* type. Otherwise, a new user that is not yet registered has entered the room. In this case, the `beliefbase` is updated accordingly.

```
getBeliefbase().getBelief("temp").setFact(usertemp);
getBeliefbase().getBelief("lightvalue").setFact(userlight);
```

Listing 4.6: Code snippet (3) of EnteredRoomPlan

When a new user enters the room, also the setpoint values of the temperature and the light intensity change. Thus, the belief base is updated with the new values (cf. Listing 4.6). In case of more than one user being in the room, the setpoint values need to be aligned. Such a mechanism based on a simple mean value calculation is also implemented here.

```
gma = get_agent_addr("delivers␣global_view");
asa = get_agent_addr("delivers␣bas_interface");
    ...
stok = get_data( "local_temp", asa);
```

```
getBeliefbase().getBelief("actemp").setFact(Double.parseDouble(stok
    .nextToken()));
```

Listing 4.7: Code snippet (4) of EnteredRoomPlan

For its operation, the room agent maintains communication relationships with other agents. In the plan, the address information of these other agents must be obtained. The `get_agent_addr()` method dispatched a new goal to retrieve the requested address of the agent. The goal (`df_search`; cf. Listing 4.3) is achieved with the help of the Jadex directory facilitator service. With the address, data can, for example, be requested from the automation system. The code for obtaining the current temperature from the automation system is shown in Listing 4.7.

```
  /* desired temperature < current temperature --> cooling) */
  if((Integer)getBeliefbase().getBelief("temp").getFact() < ((
      Integer)getBeliefbase().getBelief("actemp").getFact()-2)){
    if(outt < (Integer)getBeliefbase().getBelief("temp").getFact())
        {
      freecooling();
    } else {
      println("start air condition");
      set_action("set_aircond", "on");
      getBeliefbase().getBelief("ACon").setFact(true);
    }
  }
}
```

Listing 4.8: Code snippet (5) of EnteredRoomPlan

When the agent's beliefs are updated, the control strategies are implemented in the plan. Following the goal-plan hierarchy, different subgoals would be dispatched at this stage. In the prototype implementation, a simple temperature control algorithm is directly encoded in the plan. The algorithm in Listing 4.8 compares the desired temperature (as identified in a previous step) with the current indoor temperature of the room (as requested from the automation system agent). In the present case, the smart home system implements two cooling options. First, it is possible to open the windows (`freecooling()`), second also an air-conditioning system ("ACon") is installed. The selection of one action depends on the current outdoor temperature and the current state of the air-conditioning system. If it is colder outside, free cooling is used. Otherwise, the air-conditioning equipment is activated.

```
private void set_action(String action, String function){
  IGoal tw = createGoal("rp_initiate");
  tw.getParameter("action").setValue(action+" "+function+" "+
      getBeliefbase().getBelief("roomname").getFact());
  tw.getParameter("receiver").setValue(this.asa);
  try
  {
    dispatchSubgoalAndWait(tw);
  }
  catch(GoalFailureException gfe)
  {
    String errormsg = action+" set to "+function+" was not possible";
    println(errormsg);
  }
}
```

Listing 4.9: Code snippet (6) of EnteredRoomPlan

Listing 4.9 explains the method `set_action()` in detail. This method is used whenever the operation mode of an equipment or device shall be changed. The `createGoal(type)` method creates a goal from a template goal of the `type` kind, where the goal type must be defined in the ADF file of the agent. In the example, a "request plan" (according to the FIPA request protocol) goal is created. Subsequently, the goal parameters are set. For example, the `receiver` parameter is set to the automation system agent (`this.asa`) which is the responsible agent for the execution of the actions in the environment. When fully composed, the goal is dispatched and its execution is awaited (`dispatchSubgoalAndWait(tw)`) as well as possible exceptions are handled. Internally, Jadex waits for the subgoal to finish and automatically drops it when successful. The waiting time before the goal fails can be set as optional second parameter of the dispatch method.

```
IMessageEvent replymsg = getEventbase().createReply((IMessageEvent)
    getReason(), reply);
replymsg.getParameter(SFipa.CONTENT).setValue(cont);
sendMessage(replymsg);
```

Listing 4.10: Code snippet (7) of EnteredRoomPlan

At the end of the plan body, a message event is created to inform the initiator of the plan of its success or failure. In the depicted case (cf. Listing 4.10), the user entering the room triggered the plan execution so the recipient of the message is set to the associated user agent. This is done by creating a reply message ("replymsg")

using the `createReply()` method. The message again follows the FIPA standard. The content is set to the value of the "cont" parameter. This variable was set already during the plan execution according to the success or failure of the previous plan steps. Finally, the message is sent to the user agent.

4.3 Evaluation

The presented smart home system was prototypically implemented in the Jadex BDI agent framework, using Java as programming language. For the prototype of the smart home system, in total more than 40 different plans and almost as many communication actions were implemented in the five different agents.

Generally, all Jadex plans have similar structures and complexity, which is also true in the smart home system. However, the variation of the plans is found in the fact that they implement different control approaches and communication protocols. On one hand, for the control strategies, selected algorithms were implemented that showcase different possibilities in the smart home. The conceptual work of their design as well as their practical realization was supported by a master student. Many details on the implementation can therefore be found in the student's master's thesis (cf. Bartl in [127]). In particular, in most cases a standard control option (e.g., thermal comfort using the traditional heating/cooling system)) and a more advanced control approach (e.g., thermal comfort using free cooling) were implemented that are furthermore characterized by different energy conservation possibilities. Naturally, control strategies differ in their complexity, a fact which is also reflected in the source code of the plans. On the other hand, the communication protocols in the smart home are similar from an implementation point of view, but have differences in the associated communication partners as well as the transmitted information.

As a final step, the newly developed multiagent system needs to be tested for its functionalities and its applicability and usefulness in the smart home case. The first aspect, i.e., the availability of required functionalities, can be checked by reviewing the created programm code and comparing it to the specifications that were developed in the beginning. This step was already accomplished in course of the extensive description and discussion of representative implementation details in the previous section. Although the implemented prototype does not cover all functionalities, control possibilities and scenarios, it provides a realization of a vertical subset of the overall system functions. Thus, it implements selected functions of all system domains, ranging from sensing and acting in the environment over agent communi-

cation and messaging to core smart home functionalities, such as temperature and device control.

The second main point is the evaluation of the system performance in terms of the adequacy of its decisions and control actions. The main question to be answered is how well the system performs under real world conditions and whether it is capable of doing so at all. Obviously, the smart home system is set in a complex environment, the smart home. Main aspects of this environment are objects (mainly equipment and building structure), humans (the human users) and the physical characteristics and values of this environment. Especially the latter points introduce most of the complexity in the smart home evaluation. However, it are exactly the physical parameters of the world (e.g., temperatures or humidity) and the user (his/her mental attitudes) that define the major success criteria of the system. Additionally, energy efficiency can be regarded of the cost function of the smart home system.

The challenge is that the key criteria listed above are too complex and have too complex interdependencies to capture them sharply and in full details (either through sensing equipment or in a computer model). This is not only true for the physical parameters but even more for the user aspect. For example, capturing the "user experience" [128] is a process that requires knowledge in further domains, such as human physiology, psychology, and even sociology. However, a smart home system must perform under these conditions so that an evaluation must include them at least in a simplified way.

Thus, for an evaluation of the system performance, two approaches exist. First, the prototype system can be installed in a real world smart home and the success criteria can be measured with corresponding tools and instruments. While this is technologically feasible for the physical parameters, the methods for capturing the user attitudes are a research field for their own. The special methods need to be tailored to the human user and his/her feelings for example regarding the provided comfort. Such an evaluation approach of the subjective aspects was researched and presented by Mahdavi and Unzeitig in [129].

An organizational shortcoming of the real world test setting are the time horizons of the environmental processes and the control strategies that influence them. Due to the long inertia of, for example, an air temperature change, also long observation times need to be calculated which considerably complicates the execution of a test process. An additional drawback of the real world evaluation stems from pragmatic issues. Smart homes are not yet in widespread use and related equipment is expensive in acquisition and configuration. Thus, finding a suitable smart home that is

furthermore usable over a longer period of time and for research purposes is hard to achieve in times of limited funding.

The second evaluation option is simulation. Simulation alleviates some of the shortcomings of a real world test scenario, but introduces other restrictions and limitations. First of all, simulation has the benefit of being independent in time, place and type of the smart home. Hence, buildings of all sizes and equipment can be tested. Also safety or security critical situations that might arise in course of program errors do not pose real threats to people or infrastructure. Additionally, the environment of the smart home can be defined as desired and changed instantly without effort. Another benefit concerns the timing problem of the physical processes. Time is relative in simulation environment (and not only there). In simulation, a separate time base can be established so that time consuming processes can be executed within fractions of the real time. In other words, the simulation time is flexible and can be used to speed up the effects of physical interactions. Finally, since simulation completely occurs in a computer environment, data logging and data analysis are facilitated since no dedicated sensor equipment is needed. Main drawback of simulation is the aforementioned complexity of the physical environment and its processes. In the simulation, a model of the world (or at least of the relevant parts of the environment) is needed that in some way emulates the real environment.

Summarizing, simulation was identified as the most appropriate means for an evaluation of the proposed system. Although the implementation of the environment model might demand a partly significant abstraction of the complex world details in the present case, simulation offers first valuable results and feedback for further development quickly and can subsequently be extended as needed. Furthermore, it is important to keep in mind that a first assessment of the overall smart home system and not of any of its specific details (such as single control strategies) is sought. This is best fulfilled by simulation.

Currently, no dedicated simulation tool for smart homes is available that could be used with reasonable effort for the evaluation. Although several approaches exist that cover aspects related to the smart home (e.g., EnergyPlus which offers a simulation for building performance [130] or simulation used for assessment of building operation [131]), these are of limited use in case of the multiagent system evaluation.

For these reasons, a basic simulation approach that is particularly dedicated to the use in a multiagent system was conceptually designed and a first implementation was performed. The main idea behind the simulation approach is to transparently integrate it with the multiagent smart home system. Therefore, the concept is to

encapsulate the simulation functionality within an additional dedicated agent, the so-called *simulation agent*. As shown in Figure 4.16, the simulation agent is equipped with a simplified world model of the smart home environment. It allows the simulation of the effects of actions on the environment as well as it simulates changes in the environment that may influence the smart home system operation.

The natural access point of the simulation agent in the present system is the system's main interface to the smart home environment. This environment interface is realized by the automation system agent which can be thought of a broker between environment and control system. As depicted, during normal system operation, the automation system agent communicates directly with the underlying automation systems. In case of simulation, the automation system agent communicates with the simulation interface. Thus the simulation agent is conceptually modeled to operate behind the automation system agent thereby substituting the real environment with a simulated one. Viewed from the direction of the other smart home agents, the automation system agent abstracts the environment as it provides generic access to data values. In case of the evaluation, the environment is represented fully by the simulation agent which generates (i.e., simulates) the require data. The same approach applies to the case of communication of the smart home agents with the environment. In this direction, the automation system agent now forwards the data to the simulation agent which interprets the sent data and adapts its internal environment model accordingly. It is important to note that no changes were made to the automation system agent. Rather, the same generic internal interface is used that also enables communication with the concrete automation systems[2].

Presently, the implementation of the simulation agent features thermal comfort aspects, basic device models and an estimation of the energy consumption. Taken together, these are considered the central aspects of the smart home environment. Additionally, the simulation agent provides a simulation time to the system and thus allows to exploit the speed gains of a simulation approach.

The simulation agent is implemented as any other agent of the smart home system. Therefore, it also has an ADF file where its goals, beliefs and plans as well as its communication possibilities are specified. The central goal of the agent is to simulate the environment, which is a permanently active *achieve goal*. Most of the simulation agent functionality is implemented in one main plan that fulfills the "simulate" goal.

[2]Obviously, there is a need to implement specific drivers for each underlying automation system that map the communication principles of the systems. The communication with the simulation agent directly uses the generic interface, i.e., there is no need for a particular driver in case of simulation.

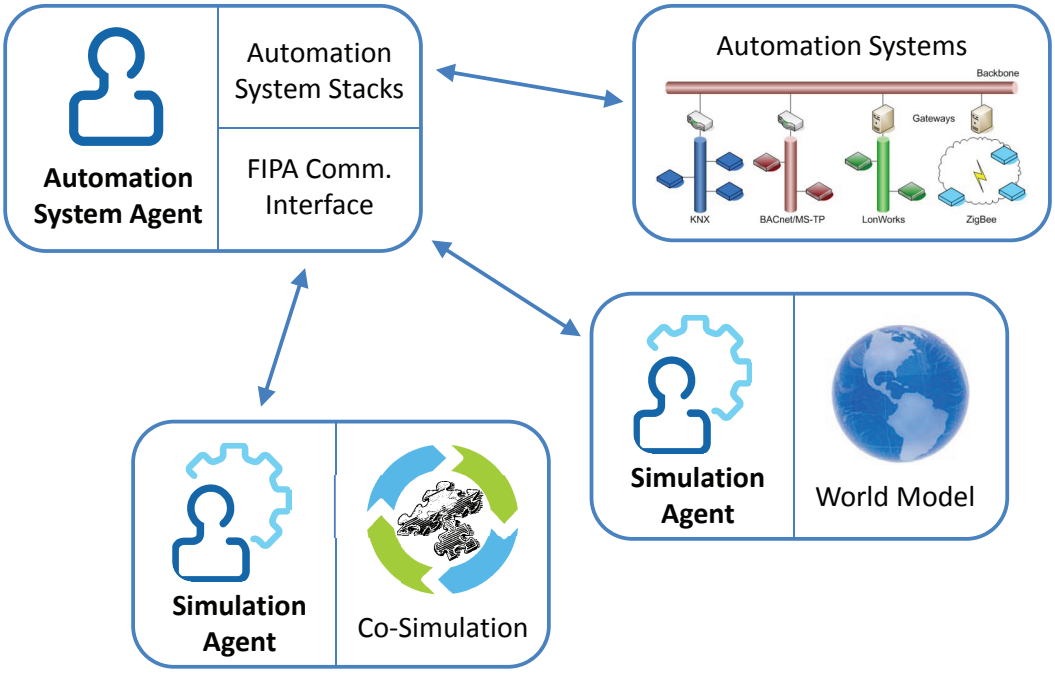

Figure 4.16: Simulation possibilities and world interface of the smart home

This plan implements typical outdoor temperature and outdoor light intensity curve for one day. This is done by interpolation of the curve based on a predefined set of typical values for a spring day (i.e., a temperature range from 12 to 22 degrees Celsius). At each discrete time step of the simulation, the current values are calculated and transmitted to the automation system agent that in turn provides it to the agent society of the smart home as requested. The time step granularity of the simulation is configurable and is initially set to 1000 seconds (i.e., approx. 16 minutes). In a similar manner, also a curve of locally produced energy is created. This information can be used, for example, to test the energy efficient scheduling of appliances in the smart home. Again, the available energy amounts are communicated to the automation system agent.

Then, the plan implements functions that are invoked if a room agent sends a heating or cooling command to the automation system agent. The latter agent normally executes the action in the environment, which reacts accordingly. This reaction is simulated by an algorithm that adds or subtracts a constant value to/from the current room temperature at each time step (i.e., in the simulation, each cooling/heating option performs equally well concerning the time needed to change the

temperature for e.g., one degree). One exception is the "natural cooling/heating" control option that works by opening windows. To support this mechanism, the simulation implements a method that gradually adjusts the room temperature to the outdoor temperature. At the end of the temperature function, the simulation agent posts the newly calculated room temperature to the automation system agent.

The calculation of the amount of "free" energy follows a similar algorithm. If energy is consumed by an appliance, the amount of available energy is reduced, while optionally produced local energy is added. Another plan of the simulation agent is used to establish the simulation time base in the smart home system. The simulated time is calculated and propagated to all agents so that a consistent time base is guaranteed.

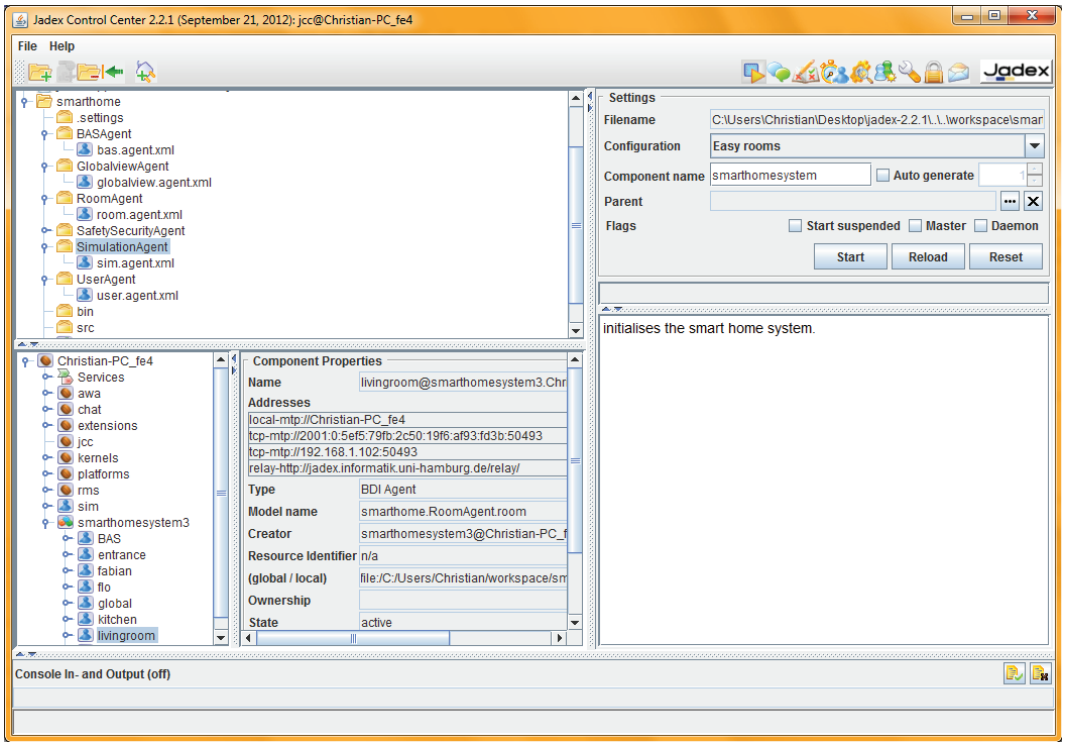

Figure 4.17: Jadex Control Center – Starter interface

Additionally to the simulation agent, the evaluation considers several events that can occur in the smart home. The creation of these events is supported by the management interface that is provided together with the Jadex framework. It is called *Jadex Control Center* (cf. Figure 4.17) and mainly provides agent management services (agent creation, dismission) as well as an overview of the currently

instantiated agent system. The control center also offers basic agent messaging functionality which allows to send miscellaneous (self-created) FIPA-compliant messages to agent instances during runtime. The so-called *Conversation Center* is shown in Figure 4.18. In course of the evaluation, this message functionality is exploited to manually send messages that notify agents about events in the smart home. Thus, such less frequent events that would normally be detected by sensors (and automatically forwarded to selected agents as messages) are simulated manually. The events simulated in that way mainly include user actions, for example, a new user entering the home or an inhabitant leave/entering a room.

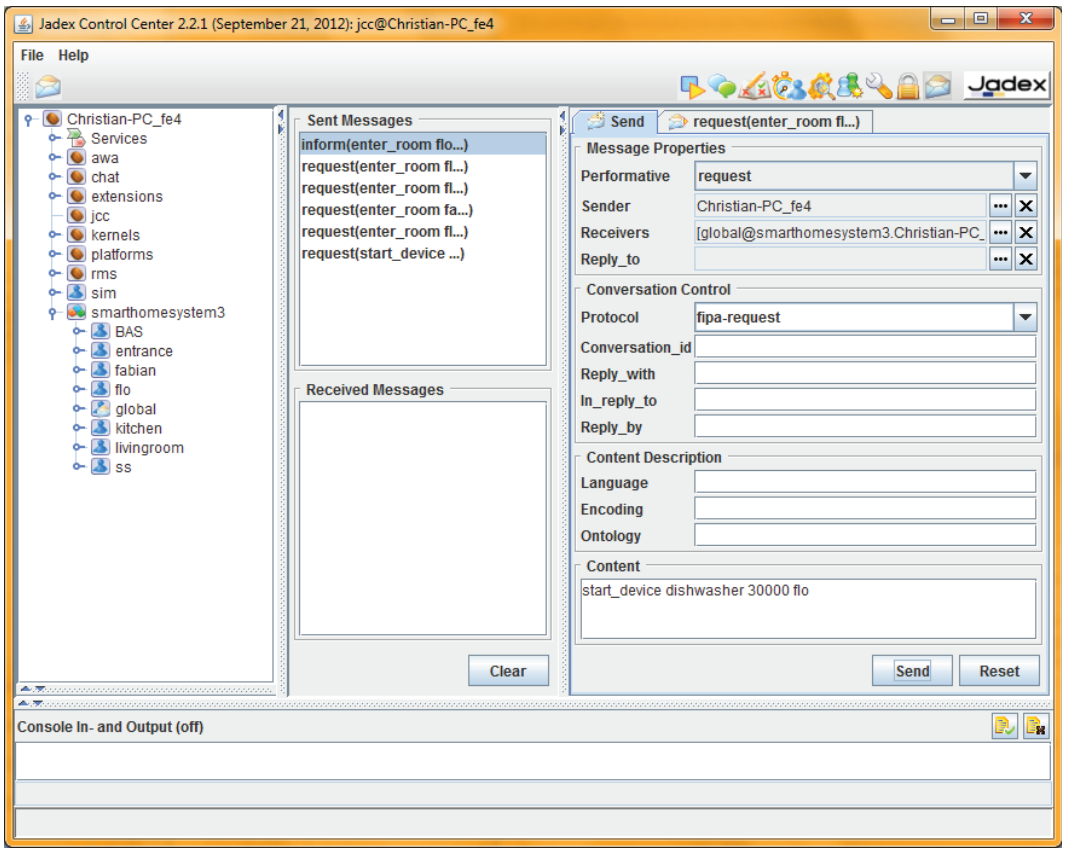

Figure 4.18: Jadex Control Center – Conversation interface

Based on the implementation of the simulation agent and supported by the event messaging of Jadex, several aspects and functions of the smart home system were evaluated. First of all, the global goals of the smart home system, energy efficiency and user comfort are tested. For this purpose, the simulation is used. In this

evaluation approach, comfort is defined as a set of user defined values for physical parameters that shall be reached during the system operation. These values are the preferred room temperature of a user and his/her light intensity preferences. The success criteria of the test are defined as the indoor temperature and light intensity reaching the comfort values. In practice, comfort should be established within reasonable time. Since no exact physical world model is available in the simulation agent, it is assumed that the equipment/processes used for temperature and light intensity changes is always capable of fulfilling the time constraints[3]. During the test it shows that the system reacts to changed setpoints without notable delay and that the simulated environment parameters reach the comfort values. Comfort is also treated in a preferred way in the system as the system always strives to achieve the user comfort. However, when different control options exist to reach the user comfort level always the most energy efficient option is chosen. This leads to the energy efficiency considerations.

In order to test the energy efficiency of the smart home system, the control approaches are evaluated. Again, temperature and light intensity are selected as observed parameters. For their control, two control possibilities are implemented for each parameter. One approach follows a conventional (i.e., energy intensive) strategy and another one implements an energy efficient strategy. For the temperature control, these control options are the use of the conventional heating/cooling system and the exploitation of free heating/cooling approaches, respectively. Hence, energy efficiency in the smart home is defined as the automatic selection of the most energy efficient control strategy considering the current world state. In the evaluation, this selection can be observed simultaneously with the comfort case. Whenever comfort is reached, the debug output shows that energy efficient options were preferred and selected whenever permitted by the world state (e.g., when there is a cooling requirement and the outdoor temperature is higher than the new room setpoint temperature value, free cooling is not applied). These approaches also help to evaluate the general decision capabilities of the smart home system. Among a set of plan, the most applicable one is selected based on a range of different criteria, in this case the state of the environment.

[3]Given the design and construction process of a smart home this is not a severe limitation. Typically, building equipment is selected based on criteria, such as the room volume, building insulation, and similar aspect. Thus, equipment that is capable of e.g., heating a room within reasonable time following a manual control approach can be expected to achieve the same performance in a smart home control scenario.

The execution of actions in the environment is evaluated with the help of the simulation agent. As this agent simulates the environment, it must receive according commands (or messages in the agent context) from the automation system agent. This central characteristic of the smart home system is already shown by the two preceding examples, because environmental parameters can be influenced by actions originating in the agent system.

A central characteristic of the smart home system is its reactivity to the environment. This aspect is evaluated using the messaging capabilities of the Jadex control center. With its help, event messages are manually generated during the regular smart home operation (cf. Figure 4.18). The reaction of the system is then observed via the debug output. Reactivity is fulfilled if the system reacts appropriately to an event that occurs in its environment. In the evaluation, the cases of a user returning home and users moving around in the house are created. The corresponding messages also include changes in the preferences that are communicated to the system simultaneously (i.e., within one message). The analysis of the debug output as well as of the simulated values (in particular temperature) shows that the system detects these events and is capable of acting in accordance to them. For example, the new comfort values are entered in the belief base and control strategies are adapted accordingly.

Finally, with the message center, another event is created to test both reactivity and proactivity of the system. The sent messages contain information on deferrable appliances that are ready to be started immediately but need only finish their operation before a user defined deadline. These events are used to evaluate the scheduling capabilities of the system. Success is defined as a combination of the appliance completing its operation before the deadline and a minimized energy consumption for this operation. For this purpose, the simulation implements a function that generates a typical daily profile of available locally produced energy. In the present scenario, the availability of a photovoltaic installation in the smart home is assumed so that the simulated energy production can be coupled to the outside light intensity. The expected operation of the smart home system is that it schedules the start of the appliance to a point in time when it expects enough energy to be available or to start the appliance using external energy to meet the user defined deadline. In the evaluation, the debug output is again analyzed after the “appliance ready” message is sent. Depending on the local energy production, the appliance either starts when enough local energy is available or just in time. Thus, the appliance operation is in both cases finished before the deadline. Additionally, it can be seen that energy is

conserved as the system waits as long as possible for renewable energy and drops the respective goal only when the deadline nears.

While the above evaluation approach targets mainly the overall characteristics of the agent based smart home system, another aspect of the smart home, namely the control strategies that are embedded in the system, can also be tested using simulation. For a control strategy that is based on Artificial Neural Networks (ANNs), this was already shown in the research publication [132]. This evaluation is also of high relevance to the proposed system as the control strategy can be easily included as one further control approach in the system. In order to realize energy efficiency and comfort simultaneously, one possible approach is to calculate optimum start/stop times of the HVAC system. The basic idea that is exploited in this way is that user comfort values need only be reached when the user really occupies a room. Thus, if the system knows the expected occupancy (e.g., from occupancy profiles), it is most energy efficient to start the heating/cooling system as late as possible but in time so that the room has reached comfort values when the user enters. The implementation of the control strategy uses an ANN to gradually adapt the start/stop time to the room, i.e., it learns the thermal behavior of the room. Input parameters are the current and desired room temperature as well as the outdoor temperature. The output of the ANN is the time that is needed to reach the values.

For an evaluation of the AI based control strategy, again simulation but with a different world model is used. As simulation model the ATplus library [133] is favored, since it provides a validated model of the thermal building characteristics which works together well with the tested control strategy. Furthermore, ATplus can be used with a graphical editor, the Dymola simulation software. The simulation results show that (once the ANN is trained, which can be accomplished both online and offline) significant amounts of energy, about 2.8% of the total heating energy, can be conserved following this simple approach. This motivates the implementation of this and similar control approaches in the smart home. The smart home system proposed in this dissertation offers a complete framework with all required functionalities to quickly integrate such control approaches in the smart home.

Both presented evaluation approaches rely on simulation to allow an assessment of the system performance. As shown in Figure 4.16, the evaluation architecture is fully transparent for the agent based smart home system. Thus, it allows for an extension of the simulation part without influencing the smart home system. As ex-

plained above, a main challenge for simulation is the representation of an adequate world model. The models must be tailored to the simulation case, a limitation which for example shows in the two different models (the newly created one as well as the ATplus model) that were described and used in this dissertation. For complex systems that address different disciplines (e.g., the smart home system), defining a unified comprehensive model that covers all aspects (e.g., thermal aspects, energy distribution, weather, human users, and many more) in sufficient detail is practically infeasible. An approach that promises a solution to this problem is co-simulation. In this case, different simulation models (and tools) are coupled together with the help of a dedicated co-simulation framework. The advantage is that for each discipline a tailored simulation approach can be used. In practice, each tool still simulates parts of the problem frame while the results are organized in a timely and organizational manner by the co-simulation framework. The probably best known co-simulation framework is the Building Controls Virtual Test Bed (BCVTB) [134] which offers support for many established simulation tools, such as EnergyPlus [130], Modelica [135] or Matlab [136].

The main contribution of co-simulation in the smart home system case is its potential to increase the energy efficiency of the smart home through a holistic consideration of the problem field. A related project that shows the exploitation of co-simulation in a production facilities setting is presented in [137]. In the smart home, this approach can be used as a role model for a similar pervasive integration of simulation.

Chapter 5

Conclusion

In this dissertation, a comprehensive system concept for smart homes was presented. This system pervasively addresses the requirements and challenges smart homes and their inhabitants are facing today. Main contribution of the presented work is the specification of a smart homes system concept that considers the requirements of smart homes already from the early design stages on. The result is an open software framework that allows the implementation and execution of different, both established and novel, control approaches that have the potential to solve the current challenges in this domain. Furthermore, this dissertation features an extensive analysis of intelligent building and smart homes, introduces a new definition of a smart home and comprehensively illustrates the capabilities and benefits of a multiagent system approach in the smart home case.

During all stages, the development of a comprehensive system that considers as well as addresses different aspects and challenges of smart homes was pursued. Particular attention was given to user comfort and energy efficiency, which were identified as the global system goals. Thus, the system was constructed around these goals and aims at ensuring them at all times. For this purpose, methods from the artificial intelligence domain, control theory and knowledge management were brought together in the system and their cooperation was enabled. For the realization, a multiagent system was identified as best suited means. Such a system was developed adhering to a methodological approach that started with the identification of challenges, requirements and use cases, was followed by an extensive architecture definition and concluded with a prototype implementation and a first validation.

The present dissertation is the result of several years of research in the domains of building automation, home automation, smart homes and adjacent areas, such

as knowledge engineering or security considerations. Out of this research, several publications at international conferences and journals emerged. These events and in particular the received reviews as well as the discussion with fellow researchers at the conferences were continuously integrated into the smart home system development process. Thanks to them, the system gradually evolved into its present form, could be tailored to satisfy requirements from many domains (e.g., smart grids) and finally can be considered as a basically validated approach.

The publications with relevance to the smart home system can be grouped into four main categories. First, the overall system concept was developed and scientifically disseminated. Its main parts with a focus on the interfaces to related disciplines were therefore presented in [138]. A more detailed description of the smart home system illustrating the three main system parts (knowledge base, multiagent system and embedded control strategies) was given in [124]. This work also features a first outlook on the system architecture as well as an overview of the prospective system overview. Starting from the overall system architecture, three further general research lines were pursued. On one hand, the abstract system architecture was instantiated as a multiagent system. Details on the according research steps, a description of multiagent systems and argumentation for their exploitation in the smart homes were provided in [139]. In addition to the multiagent approach towards improved smart home control, selected system design artifacts were described and a first overview of the agent types in the smart home were presented there. On the other hand, extensive work was performed in the area of the internal system functionalities, in particular concerning the hosted control strategies as well as the complementing knowledge base. Concerning the knowledge base, details on its different conceptual parts can be found in several publications. A first general outlook on the benefits of knowledge representation in the smart home can be found in [140], while a more energy-related review of the benefits was published in [141]. The concrete parameters for an energy-efficient smart home operation including a novel storage concept for them was presented in [142]. A comprehensive description of energy related parameters in the smart home, the storage and their integration in novel use cases enabled by the multiagent approach were illustrated in [143]. Finally, a concept for the integration of external information, in particular weather information, was presented in [144]. As a fourth and concluding pillar of research and publication work, the development of novel, intelligent control strategies can be identified. All control strategies are geared towards the exploitation of the features offered by the smart home system as described in this dissertation and mostly im-

plement the artificial intelligence in the smart home. In this context, research on the specification, generation and exploitation of user and usage profiles was conducted. The concept of profiling and its exploitation to provide user comfort and energy efficiency in the smart home were analyzed in [145]. Finally, the relation between smart homes and the energy market as well as possible approaches to use profiles for a more economic and ecological control in the smart home were published in [146].

The publications demonstrate that the presented comprehensive system for the use in smart home control is considered a valid approach by the scientific community. In the publications as well as the evaluation approach, it was shown that the system works in accordance with user satisfaction and energy consumption guidelines. It was furthermore shown that the proposed system is capable of realizing improved control in the smart home, thereby fulfilling the initially specified and constantly targeted requirements in a best effort way. The smart home control system already allows the control of a variety of building services in its presented form and evolution stage. However, the flexibility introduced by the system's realization as a multiagent system allows an easy extension of it in different aspects. Future extensions could therefore be made concerning the control strategies. Their functionality is encapsulated in dedicated agents so that a virtually unlimited range of control strategies can co-exist in the smart home system. These may incorporate further data in their control approach, provide control over further smart home functions and could also realize control functionalities at a district or city level, for example the integration into smart grids and smart cities.

One particular future extension that is already well supported by the proposed system is model predictive control, as for example described in [147]. The coupling of the smart home system with a simulation part (i.e., the simulation agent) allows to simulate different control options before a particular action is executed in the (real) environment. Following this approach, it becomes possible to evaluate control strategies regarding miscellaneous criteria, such as applicability or efficiency during runtime and thus allows to realize further control optimizations. Although the smart home system architecture already provides the required interfaces and basic services to enable model predictive control, the simulation part of the system needs refinement and extension so that better control performance can be achieved. Given the complexity and heterogeneity of the simulated environment, the development of such an extended simulation part can be challenging. One possible way to tackle especially the heterogeneity of the environment that shows for example in simulation time granularity (milliseconds for device or lighting control opposed to simula-

tion horizons of minutes for thermal building aspects), is the use of a co-simulation framework. There, different simulation environments, with each simulation possibly tailored to a particular domain, are coupled together into an execution framework. The co-simulation framework, for example the Building Controls Virtual Test Bed (BCVTB) [134], takes care of the simulation execution with respect to the different time constraints as well as provides the necessary services for data exchange among the simulation systems. Its main benefit is that it enables to use the best suited simulation systems for each simulation task. This results in confident choices of the best applicable strategies which in turn can be leveraged in an improved energy and comfort performance.

Finally, the proposed multiagent system is not limited to the use in smart homes. Although the system is currently tailored to the requirements of smart homes, it is – with adaptations – applicable also to other domains. Clearly, the use of the system in the related domain of (commercial) buildings and offices is possible with manageable effort. However, the multiagent approach can be beneficial also on higher scales, for example, to manage energy distribution (local production and consumption) at area or district levels. The general approach also bears the potential to be extended to the control of smart cities, where agents could automatically orchestrate resource needs and resource offers in a dynamic and adaptive way.

Bibliography

[1] International Energy Agency (IEA), "Energy Technology Perspectives: Scenarios and Strategies to 2050 – Fact Sheet Buildings and Appliances," 2006.

[2] Statistics Austria, "Overall energy balances Austria 1970 to 2010," accessed 10/12/2012. [Online]. Available: http://www.statistik.at/

[3] ——, "Energy consumption of households as of 2003/2004," accessed 10/12/2012. [Online]. Available: http://www.statistik.at/

[4] E-Control, "Wissenswertes zum Thema Energieverbrauch," accessed 10/12/2012. [Online]. Available: http://www.e-control.at/de/konsumenten/energie-sparen/thema-energieverbrauch/

[5] International Energy Agency (IEA), "World Energy Outlook," 2011.

[6] W. Kastner, G. Neugschwandtner, S. Soucek, and H. M. Newman, "Communication systems for building automation and control," *Proceedings of the IEEE*, vol. 93, no. 6, pp. 1178–1203, 2005.

[7] M. Brambley, P. Haves, S. McDonald, P. Torcellini, D. Hansen, D. Holmberg, and K. Roth, *Advanced sensors and controls for building applications: Market assessment and potential R & D pathways.* United States Departement of Energy, 2005.

[8] W. Kastner, S. Soucek, C. Reinisch, and A. Klapproth, "Building and Home Automation," in *Industrial Electronics Handbook*, 2nd ed., B. Wilamowski and J. Irwin, Eds. CRC Press, 2011, vol. 2: Industrial Communication Systems, ch. 26, pp. 26–1 – 26–15.

[9] T. Sauter, "Integration aspects in automation - a technology survey," in *10th IEEE Conference on Emerging Technologies and Factory Automation*, vol. 2, 2005, pp. 255–263.

[10] T. Sauter, S. Soucek, W. Kastner, and D. Dietrich, "The evolution of factory and building automation," *IEEE Magazine on Industrial Electronics*, vol. 5, no. 3, pp. 35–48, 2011.

[11] S. Soucek and D. Loy, "Vertical Integration in Building Automation Systems," in *5th IEEE International Conference on Industrial Informatics*, vol. 1, 2007, pp. 81–86.

[12] B. Graf, M. Hans, and R. Schraft, "Care-o-bot ii—development of a next generation robotic home assistant," *Autonomous robots*, vol. 16, no. 2, pp. 193–205, 2004.

[13] D. Dietrich, W. Kastner, and H. Schweinzer, "Perception Awareness in Automation–A New "Bionic" Approach," *Automatisierungstechnik*, vol. 52, no. 3–2004, pp. 107–116, 2004.

[14] A. Mahdavi, "Predictive simulation-based lighting and shading systems control in buildings," in *Building Simulation*, vol. 1, no. 1. Springer, 2008, pp. 25–35.

[15] F. Aldrich, "Smart homes: Past, present and future," in *Inside the Smart Home*, R. Harper, Ed. Springer London, 2003, pp. 17–39.

[16] "Home Automation Driving Data Innovation," 2012, accessed 05/08/2012. [Online]. Available: http://www.datanami.com/

[17] "ZigBee homepage," accessed 09/25/2012. [Online]. Available: http://www.zigbee.org/

[18] T. Grechenig, M. Bernhart, R. Breiteneder, and K. Kappel, *Softwaretechnik: Mit Fallbeispielen aus realen Entwicklungsprojekten*. Pearson Education, 2009.

[19] "Building automation and control systems (BACS) – part 5: Data communication protocol," ISO 16484-5, 2012.

[20] "Information technology – Home Electronic Systems (HES) architecture," ISO/IEC 14543-3, 2006.

[21] "Control network protocol specification," ANSI/EIA/CEA 709.1, 1999.

[22] "ZigBee 2007 specification," ZigBee Alliance, 2007. [Online]. Available: http://www.zigbee.org/

[23] "Wireless Short-Packet (WSP) protocol optimized for energy harvesting – Architecture and lower layer protocols," ISO/IEC 14543-3-10, 2012.

[24] "Radio frequency identification for item management – Part 1: Reference architecture and definition of parameters to be standardized," ISO/IEC 18000-1, 2008.

[25] "A.C.-supplied electronic ballasts for tubular fluorescent lamps – Control interface for "Control by digital signals"," IEC 60929 Annex E, 2006.

[26] "Standard motor interface," 2012. [Online]. Available: http://www.smi-group.com/

[27] "Networked device interoperability guidelines," Digital Living Network Alliance (DLNA), 2006. [Online]. Available: http://www.dlna.org/

[28] "Communication systems for and remote reading of meters - part 3: Dedicated application layer," EN 13757-3, 2011.

[29] R. Harper, "Inside the smart home: Ideas, possibilities and methods," in *Inside the Smart Home*, R. Harper, Ed. Springer London, 2003, pp. 1–13.

[30] S. S. Intille, K. Larson, J. S. Beaudin, J. Nawyn, E. M. Tapia, and P. Kaushik, "A living laboratory for the design and evaluation of ubiquitous computing technologies," in *CHI '05 extended abstracts on Human factors in computing systems*, ser. CHI EA '05. ACM, 2005, pp. 1941–1944.

[31] "House_n: The Place Lab," Project Homepage. [Online]. Available: http://architecture.mit.edu/house_n/placelab.html

[32] J. A. Kientz, S. N. Patel, B. Jones, E. Price, E. D. Mynatt, and G. D. Abowd, "The Georgia Tech aware home," in *CHI '08 extended abstracts on Human factors in computing systems*, ser. CHI EA '08. ACM, 2008, pp. 3675–3680.

[33] F. Martini, "T-Com House - At Home in the Future," *Pictures of the Future*, p. 20, 2005.

[34] "inHaus," Project Homepage. [Online]. Available: http://www.inhaus.fraunhofer.de/

[35] V. Grinewitschus and K. Scherer, "inHaus-2: Ein neues Konzept für die kooperative Entwicklung von Lösungen für das Betreute Wohnen," *Ambient Assisted Living Kongress*, 2008.

[36] "AMIGO – Ambient intelligence for the networked home environment," Project Homepage. [Online]. Available: http://www.hitech-projects.com/euprojects/amigo/

[37] "iHomeLab Research Laboratory for Building Intelligence," Project Homepage. [Online]. Available: http://www.ihomelab.ch/

[38] "SmartHOME II," Project Homepage. [Online]. Available: http://www.unibw.de/eit8_2/forschung/projekte/shII/index_html

[39] Y. Wang, Y. Shao, and C. Kargel, "Demand controlled ventilation strategies for high indoor air quality and low heating energy demand," in *Instrumentation and Measurement Technology Conference (I2MTC), 2012 IEEE International*, may 2012, pp. 870 –875.

[40] "Gator Tech Smart House," Project Homepage. [Online]. Available: http://www.icta.ufl.edu/gt.htm

[41] S. Helal, W. Mann, H. El-Zabadani, J. King, Y. Kaddoura, and E. Jansen, "The gator tech smart house: A programmable pervasive space," *Computer*, vol. 38, no. 3, pp. 50–60, 2005.

[42] K. Beck, *Extreme programming.* Pearson Education, 2003.

[43] M. Cohn, *User stories applied: For agile software development.* Addison-Wesley Professional, 2004.

[44] H. Kopetz, "Internet of Things," in *Real-Time Systems*, ser. Real-Time Systems Series. Springer US, 2011, pp. 307–323.

[45] O. Hersent, D. Boswarthick, and O. Elloumi, *The Internet of Things: Key Applications and Protocols*, 2nd ed. Wiley, 2012.

[46] W. Granzer, W. Kastner, and C. Reinisch, "Gateway-free Integration of BACnet and KNX using Multi-Protocol Devices," in *Proc. 6th IEEE International Conference on Industrial Informatics (INDIN '08)*, Jul. 2008, pp. 973–978.

[47] P. Palensky, D. Dietrich, R. Posta, and H. Reiter, "Demand Side Management in private homes by using LonWorks," in *Proceedings of the IEEE International Workshop on Factory Communication Systems*, 1997, pp. 341–347.

[48] C. Eastman, P. Teicholz, R. Sacks, and K. Liston, *BIM Handbook*. Wiley, 2008.

[49] A. Mahdavi, "Reflections on computational building models," *Building and Environment*, vol. 39, no. 8, pp. 913 – 925, 2004.

[50] F. Oldewurtel, A. Parisio, C. Jones, M. Morari, D. Gyalistras, M. Gwerder, V. Stauch, B. Lehmann, and K. Wirth, "Energy efficient building climate control using stochastic model predictive control and weather predictions," in *American Control Conference (ACC), 2010*. IEEE, 2010, pp. 5100–5105.

[51] X. Hong, C. Nugent, M. Mulvenna, S. McClean, B. Scotney, and S. Devlin, "Evidential fusion of sensor data for activity recognition in smart homes," *Pervasive and Mobile Computing*, vol. 5, no. 3, pp. 236 – 252, 2009, pervasive Health and Wellness Management. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S157411920800045X

[52] S. S. Intille, "Designing a home of the future," *IEEE Pervasive Computing*, vol. 1, no. 2, pp. 76–82, 2002.

[53] R. Hofer, "Untersuchung des Einflusses von Regelungsstrategien auf den Heizwärmebedarf und die operative Raumtemperatur bei unterschiedlichen Gebäudestandards," Master's thesis, Vienna University of Technology, 2010.

[54] K. Kappel, "Persuasive technology: ambient information systems for residential energy awareness," Ph.D. dissertation, Vienna University of Technology, 2009.

[55] B. Fogg, "Persuasive technology: using computers to change what we think and do," *Ubiquity*, vol. 2002, no. December, p. 5, 2002.

[56] A. Cockburn, *Writing effective use cases*. Addison-Wesley Boston, 2001, vol. 1.

[57] American Society of Heating, Refrigerating and Air-Conditioning Engineers (ASHRAE), "ANSI/ASHRAE Standard 55-2010 – Thermal Environmental Conditions for Human Occupancy," Published standard, 2010.

[58] P. O'Brien and R. Nicol, "FIPA – towards a standard for software agents," *BT Technology Journal*, vol. 16, no. 3, pp. 51–59, 1998.

[59] Y. Shoham, "An overview of agent-oriented programming," *Software agents*, vol. 4, pp. 271–290, 1997.

[60] H. Nwana, "Software agents: An overview," *Knowledge Engineering Review*, vol. 11, no. 3, pp. 205–244, 1996.

[61] S. Russell and P. Norvig, *Artificial intelligence: a modern approach*. Prentice hall, 2009.

[62] M. Wooldridge, *An Introduction to Multiagent Systems*, 2nd ed. Wiley, 2009.

[63] J. Bradshaw, "An introduction to software agents," *Software agents*, vol. 5, pp. 3–46, 1997.

[64] D. Gilbert, M. Aparicio, B. Atkinson, S. Brady, J. Ciccarino, B. Grosof, P. O'Connor, D. Osisek, S. Pritko, R. Spagna *et al.*, "IBM intelligent agent strategy – White paper," *IBM Corporation*, 1995.

[65] M. Wooldridge and N. Jennings, "Agent theories, architectures, and languages: a survey," *Intelligent agents*, pp. 1–39, 1995.

[66] M. Bratman, *Intentions in communication*. MIT, 1990, ch. What is intention?, pp. 15–32.

[67] M. Winikoff, L. Padgham, and J. Harland, "Simplifying the development of intelligent agents," *AI 2001: Advances in Artificial Intelligence*, pp. 557–568, 2001.

[68] P. Maes, "The agent network architecture (ana)," *ACM SIGART Bulletin*, vol. 2, no. 4, pp. 115–120, 1991.

[69] M. Wooldridge and N. Jennings, "Intelligent agents: Theory and practice," *Knowledge engineering review*, vol. 10, no. 2, pp. 115–152, 1995.

[70] M. Bratman, *Intention, plans, and practical reason*. Harvard University Press Cambridge, 1987.

[71] A. S. Rao and M. P. Georgeff, "BDI agents: From Theory to Practice," in *Proceedings of the First International Conference on Multi-agent Systems (ICMAS '95)*, 1995, pp. 312–319.

[72] D. Kinny and M. P. Georgeff, "Commitment and effectiveness of situated agents," in *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence (IJCAI '91)*, 1991, pp. 82–88.

[73] M. Georgeff and A. Rao, "Rational software agents: from theory to practice," *Agent Technology: Foundations, Applications, and Markets*, vol. 139, p. 160, 1998.

[74] L. Padgham and M. Winikoff, *Developing intelligent agent systems.* John Wiley, 2004.

[75] M. Georgeff and A. Lansky, "Reactive reasoning and planning," in *Proceedings of the 6th national conference on artificial intelligence (AAAI-87)*, 1987, pp. 677 – 682.

[76] M. Huber, "JAM: A BDI-theoretic mobile agent architecture," in *Proceedings of the 3rd annual conference on Autonomous Agents*, 1999, pp. 236–243.

[77] "JACK autonomous software," Project Homepage. [Online]. Available: http://aosgrp.com/products/jack/index.html

[78] M. Winikoff, "Jack^{TM} intelligent agents: An industrial strength platform," *Multi-Agent Programming*, pp. 175–193, 2005.

[79] "Jadex - BDI Agent System," Project Homepage. [Online]. Available: http://jadex.informatik.uni-hamburg.de/

[80] R. Lima, R. Sousa, and P. Martins, "Distributed production planning and control agent-based system," *International journal of production research*, vol. 44, no. 18-19, pp. 3693–3709, 2006.

[81] Q. Feng, A. Bratukhin, A. Treytl, and T. Sauter, "A flexible multi-agent system architecture for plant automation," in *Industrial Informatics, 2007 5th IEEE International Conference on*, vol. 2. IEEE, 2007, pp. 1047–1052.

[82] S. Abras, S. Ploix, S. Pesty, and M. Jacomino, "A multi-agent home automation system for power management," *Informatics in Control Automation and Robotics*, pp. 59–68, 2008.

[83] D. Cook, M. Youngblood, and S. Das, "A Multi-agent Approach to Controlling a Smart Environment," *Designing Smart Homes*, pp. 165–182, 2006.

[84] D. Cook, M. Youngblood, E. Heierman III, K. Gopalratnam, S. Rao, A. Litvin, and F. Khawaja, "Mavhome: An agent-based smart home," in *Pervasive Computing and Communications, 2003.(PerCom 2003). Proceedings of the First IEEE International Conference on.* IEEE, 2003, pp. 521–524.

[85] G. Youngblood, D. Cook, and L. Holder, "The mavhome architecture," *Department of Computer Science and Engineering University of Texas at Arlington, Techinal Report*, 2004.

[86] B. Qiao, K. Liu, and C. Guy, "A Multi-Agent System for Building Control," in *Proceedings of IEEE Int. Conference on Intelligent Agent Technology (IAT '06)*, 2006, pp. 653 –659.

[87] K. Liu, C. Lin, and B. Qiao, "A multi-agent system for intelligent pervasive spaces," in *Service Operations and Logistics, and Informatics, 2008. IEEE/-SOLI 2008. IEEE International Conference on*, vol. 1. IEEE, 2008, pp. 1005–1010.

[88] H. Hagras, V. Callaghan, M. Colley, G. Clarke, A. Pounds-Cornish, and H. Duman, "Creating an ambient-intelligence environment using embedded agents," *Intelligent Systems*, vol. 19, no. 6, pp. 12 – 20, 2004.

[89] F. Doctor, H. Hagras, and V. Callaghan, "A fuzzy embedded agent-based approach for realizing ambient intelligence in intelligent inhabited environments," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, vol. 35, no. 1, pp. 55–65, 2005.

[90] P. Davidsson and M. Boman, "Saving Energy and Providing Value Added Services in Intelligent Buildings: A MAS Approach," in *Agent Systems, Mobile Agents, and Applications*, ser. LNCS, 2000, pp. 79–143.

[91] M. Boman, P. Davidsson, N. Skarmeas, K. Clark, R. Gustavsson *et al.*, "Energy saving and added customer value in intelligent buildings," in *Third International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology*, 1998, pp. 505–517.

[92] P. Davidsson and M. Boman, "A multi-agent system for controlling intelligent buildings," in *Proceedings of the Int. Conference on MultiAgent Systems*, 2000, pp. 377 –378.

[93] K.-I. Benta, A. Hoszu, L. Văcariu, and O. Creţ, "Agent based smart house platform with affective control," in *Proceedings of the Euro American Conference on Telematics and Information Systems (EATIS '09)*, 2009, pp. 18:1–18:7.

[94] S. Abras, S. Pesty, S. Ploix, and M. Jacomino, "Advantages of mas for the resolution of a power management problem in smart homes," *Advances in Practical Applications of Agents and Multiagent Systems*, pp. 269–278, 2010.

[95] Z. Mo and A. Mahdavi, "An agent-based simulation-assisted approach to bilateral building systems control," in *Proceedings 8th International IBPSA Conference (Building Simulation 2003)*, 2003, pp. 887 – 894.

[96] N. Kushwaha, M. Kim, D. Kim, and W. Cho, "An intelligent agent for ubiquitous computing environments: smart home ut-agent," in *Software Technologies for Future Embedded and Ubiquitous Systems, 2004. Proceedings. Second IEEE Workshop on.* IEEE, 2004, pp. 157–159.

[97] Z. Wang, R. Yang, and L. Wang, "Multi-agent control system with intelligent optimization for smart and energy-efficient buildings," in *IECON 2010-36th Annual Conference on IEEE Industrial Electronics Society.* IEEE, 2010, pp. 1144–1149.

[98] H. Zhang, F.-Y. Wang, and Y. Ai, "An OSGi and agent based control system architecture for smart home," in *Proc. IEEE Int. Conf. on Networking, Sensing and Control*, March 2005, pp. 13–18.

[99] M. Wooldridge, N. Jennings, and D. Kinny, "The gaia methodology for agent-oriented analysis and design," *Autonomous Agents and Multi-Agent Systems*, vol. 3, no. 3, pp. 285–312, 2000.

[100] F. Zambonelli, N. Jennings, and M. Wooldridge, "Developing multiagent systems: The gaia methodology," *ACM Transactions on Software Engineering and Methodology (TOSEM)*, vol. 12, no. 3, pp. 317–370, 2003.

[101] J. Gonzalez-Palacios and M. Luck, "Extending gaia with agent design and iterative development," *Agent-Oriented Software Engineering VIII*, pp. 16–30, 2008.

[102] M. Wood and S. DeLoach, "An overview of the multiagent systems engineering methodology," in *Agent-Oriented Software Engineering.* Springer, 2001, pp. 1–53.

[103] S. DeLoach and M. Wood, "Developing multiagent systems with agenttool," *Intelligent Agents VII Agent Theories Architectures and Languages*, pp. 30–41, 2001.

[104] L. Padgham and M. Winikoff, "Prometheus: a methodology for developing intelligent agents," *Agent-oriented software engineering III*, pp. 174–185, 2003.

[105] L. Padgham, J. Thangarajah, and M. Winikoff, "Tool support for agent development using the prometheus methodology," in *Quality Software, 2005.(QSIC 2005). Fifth International IEEE Conference on*, 2005, pp. 383–388.

[106] P. Bresciani, A. Perini, P. Giorgini, F. Giunchiglia, and J. Mylopoulos, "Tropos: An agent-oriented software development methodology," *Autonomous Agents and Multi-Agent Systems*, vol. 8, no. 3, pp. 203–236, 2004.

[107] J. Mylopoulos, M. Kolp, and J. Castro, "Uml for agent-oriented software development: The tropos proposal," *«UML» 2001—The Unified Modeling Language. Modeling Languages, Concepts, and Tools*, pp. 422–441, 2001.

[108] B. Henderson-Sellers and P. Giorgini, *Agent-oriented methodologies*. Idea Group Publishing, 2005.

[109] F. Giunchiglia, J. Mylopoulos, and A. Perini, "The tropos software development methodology: processes, models and diagrams," *Agent-Oriented Software Engineering III*, pp. 162–173, 2003.

[110] F. Bellifemine, G. Caire, and D. Greenwood, *Developing multi-agent systems with JADE*. Wiley, 2007.

[111] J. Thangarajah, L. Padgham, and M. Winikoff, "Prometheus design tool," in *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*. ACM, 2005, pp. 127–128.

[112] R. Miles and K. Hamilton, *Learning UML 2.0*. O'Reilly Media, Incorporated, 2006.

[113] M. Huget, "Agent UML notation for multiagent system design," *IEEE Internet Computing*, vol. 8, no. 4, pp. 63–71, 2004.

[114] Lin Padgham and Michael Winikoff, *Developing Intelligent Agent Systems - A Practical Guide*. John Wiley and Sons Ltd., 2004.

[115] M. Winikoff, "Towards making agent UML practical: A textual notation and a tool," in *Quality Software (QSIC 2005). Fifth International Conference on*. IEEE, 2005, pp. 401–406.

[116] L. Braubach, A. Pokahr, D. Moldt, and W. Lamersdorf, "Goal representation for BDI agent systems," *Programming multi-agent systems*, pp. 44–65, 2005.

[117] S. Sardina, L. de Silva, and L. Padgham, "Hierarchical planning in bdi agent programming languages: A formal approach," in *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*. ACM, 2006, pp. 1001–1008.

[118] G. Simon, B. Mermet, and D. Fournier, "Goal decomposition tree: An agent model to generate a validated agent behaviour," *Declarative Agent Languages and Technologies III*, pp. 124–140, 2006.

[119] P. Busetta, R. Rönnquist, A. Hodgson, and A. Lucas, "JACK intelligent agents-components for intelligent agents in java," *AgentLink News Letter*, vol. 2, no. 1, pp. 2–5, 1999.

[120] N. Howden, R. Rönnquist, A. Hodgson, and A. Lucas, "JACK intelligent agents-summary of an agent infrastructure," in *Fifth International Conference on Autonomous Agents*, 2001.

[121] A. Pokahr, L. Braubach, and W. Lamersdorf, "Jadex: A BDI reasoning engine," *Multi-Agent Programming*, pp. 149–174, 2005.

[122] F. Bellifemine, G. Caire, A. Poggi, and G. Rimassa, "Jade: A software framework for developing multi-agent applications. lessons learned," *Information and Software Technology*, vol. 50, no. 1-2, pp. 10–21, 2008.

[123] M. Piunti, "Programming BDI agents in Jadex," Tutorial held at Multi-Agent Systems Conference, 2008.

[124] C. Reinisch, M. J. Kofler, F. Iglesias, and W. Kastner, "ThinkHome: Energy Efficiency in Future Smart Homes," *EURASIP Journal on Embedded Systems*, vol. 2011, p. 18, 2011.

[125] L. Braubach, A. Pokahr, and W. Lamersdorf, "Jadex: A BDI-agent system combining middleware and reasoning," *Software agent-based applications, platforms and development kits*, pp. 143–168, 2005.

[126] L. Braubach, W. Lamersdorf, and A. Pokahr, "Jadex: Implementing a BDI-Infrastructure for JADE Agents," *Exp Journal*, vol. 3, no. 3, pp. 76–85, 2003.

[127] F. Bartl, "Modelling a multi agent-system for the usage in smart homes," Master's Thesis, Vienna University of Technology, 2012, to appear.

[128] A. Mahdavi, "The human dimension of building performance simulation," *Keynote Lecture at 12th International IBPSA Conference (Building Simulation 2011)*, 2011.

[129] A. Mahdavi and U. Unzeitig, "Occupancy implications of spatial, indoor-environmental, and organizational features of office spaces," *Building and environment*, vol. 40, no. 1, pp. 113–123, 2005.

[130] D. Crawley, L. Lawrie, C. Pedersen, and F. Winkelmann, "EnergyPlus: Energy Simulation Program," *ASHRAE journal*, vol. 42, no. 4, pp. 49–56, 2000.

[131] A. Mahdavi, "Simulation-based control of building systems operation," *Building and Environment*, vol. 36, no. 6, pp. 789–796, 2001.

[132] W. Kastner, M. Kofler, and C. Reinisch, "Using AI to realize energy efficient yet comfortable smart homes," in *Factory Communication Systems (WFCS), 2010 8th IEEE International Workshop on.* IEEE, 2010, pp. 169–172.

[133] R. Merz, "Objektorientierte mathematische Modellierung thermischen Gebäudeverhaltens," Ph.D. dissertation, Dissertation Technische Universität Kaiserslautern, 2002.

[134] M. Wetter, "Co-simulation of building energy and control systems with the Building Controls Virtual Test Bed," *Journal of Building Performance Simulation*, vol. 4, no. 3, pp. 185–203, 2011.

[135] M. Tiller, *Introduction to physical modeling with Modelica.* Springer, 2001, vol. 615.

[136] D. Etter and D. Kuncicky, *Introduction to MATLAB.* Prentice Hall, 2011.

[137] I. Leobner, K. Ponweiser, G. Neugschwandtner, and W. Kastner, "Energy efficient production-a holistic modeling approach," in *Sustainable Technologies (WCST), 2011 World Congress on.* IEEE, 2011, pp. 62–67.

[138] C. Reinisch, M. J. Kofler, and W. Kastner, "ThinkHome: A Smart Home as Digital Ecosystem," in *Proceedings of 4th IEEE International Conference on Digital Ecosystems and Technologies (DEST '10)*, 2010, pp. 256–261.

[139] C. Reinisch and W. Kastner, "Agent based control in the Smart Home," in *Proceedings of 37th Annual Conference of the IEEE Industrial Electronics Society (IECON '11)*, 2011, pp. 334 – 339.

[140] C. Reinisch, W. Granzer, F. Praus, and W. Kastner, "Integration of Heterogeneous Building Automation Systems using Ontologies," in *Proceedings of 34th Annual Conference of the IEEE Industrial Electronics Society (IECON '08)*, 2008, pp. 2736–2741.

[141] M. J. Kofler, C. Reinisch, and W. Kastner, "A Knowledge Representation for Energy Parameters in Sustainable Smart Homes," in *Proceedings of Third International Conference on Applied Energy (ICAE '11)*, 2011, pp. 1065–1078.

[142] ——, "An Intelligent Knowledge Representation of Smart Home Energy Parameters," in *Proceedings of the World Renewable Energy Congress (WREC '11)*, Linköping, Sweden, 2011, pp. 921–928.

[143] ——, "A semantic representation of energy-related information in future smart homes," *Energy and Buildings*, vol. 47, no. 0, pp. 169 – 179, 2012.

[144] M. Kofler, C. Reinisch, and W. Kastner, "An Ontological Weather Representation for Improving Energy-Efficiency in Interconnected Smart Home Systems," in *Proceedings of the Conference on Applied Simulation and Modelling: Artificial Intelligence and Soft Computing*. ACTA Press, 2012.

[145] F. Iglesias Vazquez, W. Kastner, and C. Reinisch, "Impact of user habits in smart home control," in *Emerging Technologies Factory Automation (ETFA), IEEE 16th Conference on*, 2011, pp. 1–8.

[146] F. Iglesias Vazquez, W. Kastner, S. C. Gaceo, and C. Reinisch, "Electricity Load Management in Smart Home Control," in *12th Conference of International Building Performance Simulation Association, Building Simulation 2011*, 2011, pp. 957–964.

[147] A. Mahdavi and C. Pröglhöf, "A model-based approach to natural ventilation," *Building and Environment*, vol. 43, no. 4, pp. 620 – 627, 2008.